

2014

GRID-BASED OUTDOOR OBJECT RECOGNITION FOR AUGMENTED REALITY

Brian Adrian Flowers
University of Rhode Island, adflowers@gmail.com

Follow this and additional works at: <https://digitalcommons.uri.edu/theses>

Terms of Use

All rights reserved under copyright.

Recommended Citation

Flowers, Brian Adrian, "GRID-BASED OUTDOOR OBJECT RECOGNITION FOR AUGMENTED REALITY"
(2014). *Open Access Master's Theses*. Paper 435.
<https://digitalcommons.uri.edu/theses/435>

This Thesis is brought to you by the University of Rhode Island. It has been accepted for inclusion in Open Access Master's Theses by an authorized administrator of DigitalCommons@URI. For more information, please contact digitalcommons-group@uri.edu. For permission to reuse copyrighted content, contact the author directly.

GRID-BASED OUTDOOR OBJECT RECOGNITION
FOR AUGMENTED REALITY

BY

BRIAN ADRIAN FLOWERS

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

COMPUTER SCIENCE

UNIVERSITY OF RHODE ISLAND

2014

MASTER OF SCIENCE
OF
BRIAN A. FLOWERS

APPROVED:

Thesis Committee:

Major Professor	Jean-Yves Hervé
	Edmund Lamagna
	Lubos Thoma
	Nasser H. Zawia

UNIVERSITY OF RHODE ISLAND
2014

ABSTRACT

Augmented Reality of an outdoor scene as a topic has gained a great deal of popularity in recent years. This work will focus on markerless hybrid Outdoor Augmented Reality (OAR) systems. In general OAR is performed through a classical statistical approach. Strong features are calculated from images of the object, located, and tracked in the scene. Gathering such features requires specialized knowledge of Computer Vision techniques; keeping OAR from finding commercial success. Model-based approaches rely less on previously gathered data, increasingly the accessibility of such techniques, but require extensive scene understanding to correctly parse the scene. The proposed model-based approach minimizes the required scene understanding allowing for augmentation of an environment with minimal input.

ACKNOWLEDGMENTS

I would like to begin by thanking my advisor, Dr. Jean-Yves Hervé, for his mentorship and guidance throughout the years. I would also like to thank Dr. Joan Peckham, for always being willing to listen and provide advice.

I am grateful to the faculty of the Computer Science department for providing a constructive learning environment. I am equally grateful for the faculty I have had the opportunity to learn from in the Mathematics, Engineering, and Statistics departments. In particular from Dr. Lubos Thoma and Dr. Ashwin Sarma I learned practical techniques invaluable in my work. From Dr. Edmund Lamagna and Dr. Natalia Katenka I learned the analytical skills necessary to understand and diagnose the problems that may occur in my results.

Finally, I must thank my family for being there with help whenever I have needed it. Most specifically, I have to thank my elder sister, Cynthia Prudence, for giving me her collection of computational books and providing me with opportunities I otherwise would have been unaware existed.

TABLE OF CONTENTS

ABSTRACT.....	ii
ACKNOWLEDGEMENTS.....	iii
TABLE OF CONTENTS.....	iv
LIST OF TABLES.....	vi
LIST OF FIGURES.....	vii
GLOSSARY.....	ix
CHAPTER 1.....	1
INTRODUCTION.....	1
CHAPTER 2.....	5
REVIEW OF LITERATURE.....	5
CHAPTER 3.....	19
JUSTIFICATION.....	19
METHODOLOGY.....	22
IMAGE PRE-PROCESSING.....	25
VANISHING POINT DETECTION.....	27
PLANE DETECTION.....	31
STORAGE MODEL.....	36
PLANE MATCHING.....	38
CHAPTER 4.....	40
RESULTS.....	40
DISCUSSION.....	62
CHAPTER 5.....	64

FUTURE WORKS.....	64
BIBLIOGRAPHY.....	68

LIST OF TABLES

TABLE	PAGE
TABLE 1. Degradation due to window inclusion.....	60

LIST OF FIGURES

FIGURE	PAGE
Figure 1. Yelp Monocle's street view.....	2
Figure 2. Game characters being displayed on top of marker cards on the Nintendo 3DS.....	4
Figure 3. Tracking the silhouette of a mountain range.....	6
Figure 4. Wire-frame realignment	7
Figure 5. A user wearing the TOWNWEAR system.....	8
Figure 6. Augmenting a scene with a handheld device.....	9
Figure 7. Localization from a panoramic image.....	11
Figure 8. Creating a block drawing from an image.....	13
Figure 9. Detection of blocks placed on a table.....	15
Figure 10. Vanishing Point grouped line segments.....	16
Figure 11. Projection of a plane and its vanishing points.....	17
Figure 12. Building Grid Layout.....	21
Figure 13. System overview diagram.....	24
Figure 14. Image pre-processing	25
Figure 15. Vanishing Point Detection.....	27
Figure 16. Plane Detection.....	30
Figure 17. The quadrilateral projection of a face of a building.....	32
Figure 18. The vanishing lines providing the largest area quadrilateral.....	32
Figure 19. Rectified bounding quadrilateral.....	33

Figure 20. Rotation of vertical lines towards x-infinity.....	34
Figure 21. A cuboid model. Rother 2002.....	36
Figure 22. The Plane Matching Component.....	38
Figure 23. Baseline line segmentation and bounding.....	43
Figure 24. Rotated line segmentation and bounding.....	45
Figure 25. Distance line segmentation and bounding.....	47
Figure 26. Plane template (windows in blue).....	48
Figure 27. Grid segmented baseline image.....	50
Figure 28. Template with cell-wise measurements.....	51
Figure 29. Grid segmented baseline image with cell-wise measurements.....	51
Figure 30. Template with width measurements.....	52
Figure 31. Grid segmented baseline image with width measurements.....	52
Figure 32. Grid segmented rotated image.....	54
Figure 33. Histogram comparison between the scene (in blue) and the model (in red).....	57
Figure 34. Histogram comparison between the scene (in blue) and the model (in red).....	59

GLOSSARY

Augmented Reality:

Is the world perceived with additional information. This thesis will focus on Augmented Reality for Visual Data, where video/camera data of the world has imagery added to the video data.

World-Space:

The world space is the real world. This thesis will refer to the World Space as being 3-dimensional, referring to height, width, and depth.

Image-Space:

The image space is the world space after it has been projected onto an image by a camera. The image space has only 2-dimensions, height and width.

Scene:

The scene is the area world-space to be augmented.

Edge:

An edge is an area in an image that has high degree of contrast. Where there is a discontinuity in the gradient of the image intensity.

Corner:

A corner is a corner in the image formed by the intersection of edges.

Feature:

A feature is a distinct aspect of the image. Features are commonly groups of pixels, edges, corners, etc.

Feature Extraction:

The act of finding and storing features from an image.

Feature Look-up:

The act of finding a particular feature in the image.

Occluder/Occlusion:

Occlusion is when an object is being partially to completely covered by another object in the image space. An Occluder is an object covering another object in the image space.

Object Recognition:

Object Recognition is the act of identifying an object in the image space as being the projection of a specific object in the world space.

Localization:

Localization is the act of identifying one's location and orientation in world space.

Fronto-parallel:

A fronto-parallel plane is one that has a constant depth throughout the entire plane in relationship to the viewer

Rectification:

The act of transforming a region so that it is fronto-parallel

CHAPTER 1

INTRODUCTION

Augmented Reality (AR) is the act of overlaying additional information to a video/view of the environment. Common examples include to overlay a marker with information for nearby restaurants on to a map on a mobile phone, projecting images and games onto a video of the current view [47], and demoing architectural changes to building before beginning construction.

There are three different approaches towards developing Augmented Reality. First is the purely position-based approach. This approach gathers information about the user's location and/or orientation from a collection of non-imaging sensors. The most used sensors being a combination of GPS, accelerometer, inclinometer, compass, and/or gyroscope. The advantage of using such an is that the raw data from the input sensors comes in very quickly and can be used with minimal processing. Mobile phone AR applications that use the phone's GPS to find local hotspots, information about the user's current area, and position-based games being the best examples [46,31].

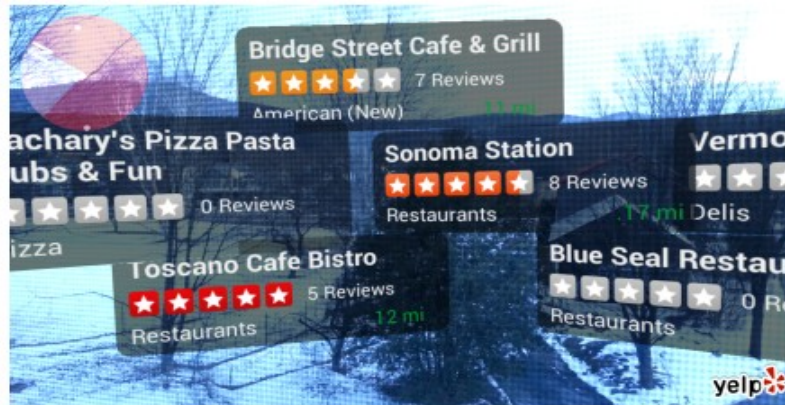


Figure 1: Yelp Monocle's street view. Angus 2013 [1]

In particular Yelp's Monocle system shown in Figure 1 is of interest. For this system, locations of interest are gathered relative to the user's GPS location and floating text boxes representing the objects are displayed on the screen based on the user's relative orientation as judged by the compass on their mobile device. As the sensors used are not precise, pure position-based systems must either be resistant to the possible inaccuracy in the data and/or have a way to better localize themselves in the world so as to order to reduce the degree of error. Due to this, such systems are not preferable when high precision is needed.

Pure vision-based systems use the data from a camera and/or video; by using computer vision techniques, these approaches tend to offer accurate frame to frame tracking, making for a more precise system. The problem with a pure vision-based approach is two fold. They require more information at start up (varying based on the

precision of the system) and requires significantly more computation time than a pure position-based approach. To overcome the lack of precision with a position-based approach, and reduce the amount of data needed to localize the vision-based approach, hybrid systems (positional sensors to assist with positional awareness combined with a vision system for precision) are now the most commonly used system for Augmented Reality [50].

As the vision component of hybrid systems will be the focus of this work, it is important to discuss the major trends in Vision-based Augmented Reality. The Vision component of an AR system either uses markers or is markerless. A marker is a uniquely identifiable pattern that can be physically placed in the scene by the user. When the vision algorithm finds the marker it can use it to localize itself in space [16,50,9]. Using marker-based AR required only setting the marker in the scene for the algorithm's initialization and calibration phase. Due to this ease of use marker-based Augmented Reality is used commercially. The latest hand held video game consoles from Nintendo and Sony, the Nintendo 3DS and the Playstation Vita respectively, both have an on-board camera and come with a set of cards to be placed in the environment to act as markers for Augmented Reality games [28,13] as shown below in Figure 2.



Figure 2: Game characters being displayed on top of marker cards on the Nintendo 3DS. Metrowebukmetro [28]

Markerless approaches are precise and do not require the user to manipulate the scene in any way (such as by adding previously created markers). However, markerless algorithms must generate and find an alternate set of recognizable features in the scene. This work will focus solely on markerless hybrid augmented reality systems when used outdoors, a category of problems also known as Outdoor Augmented Reality (OAR).

CHAPTER 2

REVIEW OF LITERATURE

Due to the scarcity of applied work focused on Outdoor Augmented Reality, this review will also cover architecture-focused object recognition, registration, and model building.

Hybrid Sensor Systems for Augmented Reality have been actively researched in an applied setting since the late 90s. When used in an indoor scene sensors would be used to help localize and reorient the user's camera in the world space. At this point, features previously gathered regarding the target object would be searched for in the projection of the scene onto the image. Given a successful feature look-up, new values are mapped onto the scene. In You and Neumann's system [48] a gyroscope was the chosen sensor as it provides increased invariance to rotation. When performing Augmented Reality in an outdoor scene however, this approach is lacking for a few key reasons.

The biggest difference between augmenting an outdoors environment as opposed to an indoor environment is the set of assumptions one has in about the environment. When indoors the size of the environment may be known in advance or calculated based on the room geometry [19], when outdoors such information is not

guaranteed to exist. When indoors, the lighting for the scene is consistent. When outdoors, shadows that move over the course of the day based on the position of the sun result in strong contrasts in the image generating new corners, blobs, and lines, which negatively affects the reliability of object recognition algorithms [50]. Additionally, when indoors a user has control over their environment, as such they may be able to place markers to be used for localization. Users are less likely to be able to modify their environment, making marker placement a significantly less reliable option when outdoors.



Figure 3: Tracking the silhouette of a mountain range. Behringer [4]

One of the first Augmented Reality systems that focused on the unique problems present in an outdoor scene was the system designed by Behringer in 1999 [4]. This system focused on the issue of distance and relative positioning. Behringer's

system was equipped with a GPS and an inclinometer. The unique aspect of this system was the silhouette-based recognition algorithm. The system worked by calculating a silhouette map formed from the mountain peaks detected in the scene. The silhouette map from the scene was then best fit against the expected silhouette of the known mountain peaks for the area. Matching the silhouette found with the predicted silhouette allowed the algorithm to fine tune its rotation. Figure 3 shows an example silhouette mapping from this algorithm. The system however had strong constraints regarding its usage. It required large non-cluttered peaks in the silhouettes such as mountain ranges and hills to be visible. Due to the reliance on having the peaks be clearly visible, the system was highly sensitive to occluders.

In the early 2000s, the needed sensors became smaller and increased CPU speeds allowed for faster feature matching.

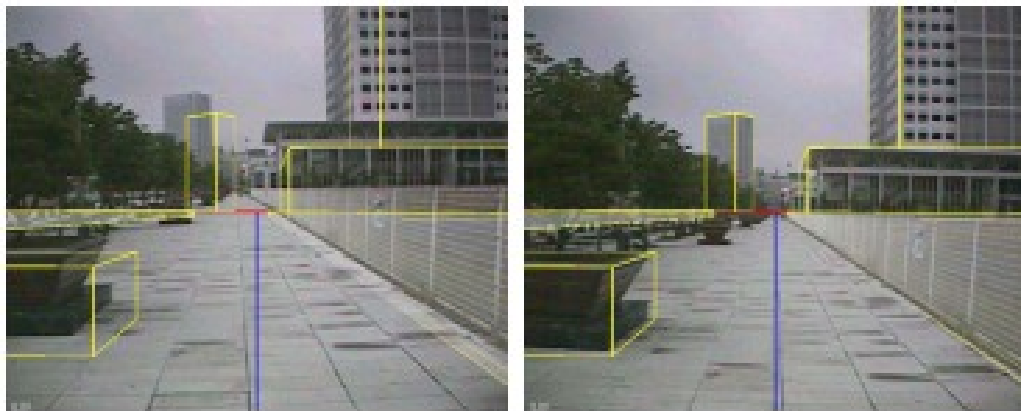


Figure 4: Wire-frame realignment. Anabuki & Yamamoto [38]

The TOWNWEAR system in 2001 side stepped the exact positioning through pre-calibration and using a set starting position [38]. The focus of the TOWNWEAR system was on calibrating the user's orientation and having a system small enough it could be worn by the user.



Figure 5: A user wearing the TOWNWEAR system. Anabuki & Yamamoto [38]

The user experienced the system through a Head Mounted Display equipped with a high precision gyroscope and a camera (Figure 5). The system worked from a single location from which the user was required to remain stationary. Certain parameters had to be manually configured by the user at initialization (and occasionally reset to account for drift). As seen in Figure 4, if the wire-frame generated for the scene did not map properly in the image space the user would reorient until the frames were a match and manually hit a key to signal the calibration. After the initial conditions were known, standard template matching was performed to look for the previously calculated defining features of the building to be found. The biggest advantage of the

system is that the entire system could be worn by a user (Figure 5). Unfortunately, the system required manual calibration before each use and the user was required to be stationary.

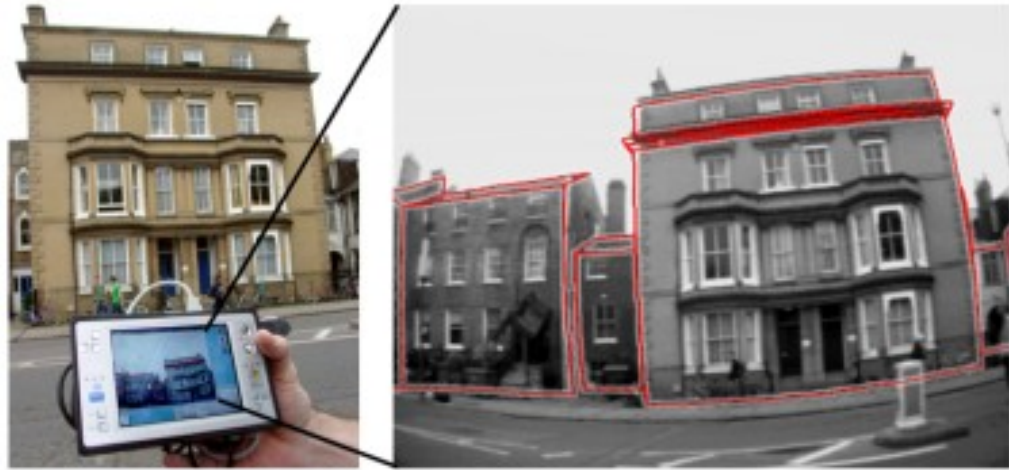


Figure 6: Augmenting a scene with a handheld device. Reitmayr & Drummond [33]

In 2006, work with hybrid Vision-Sensor systems was further refined (as shown in Figure 6). Reitmayr and Drummond's approach used and stored a textured-model of the scene as opposed to the more detailed edge model of the scene as had been common practice [33]. This system had the advantage of deciding on the edge features to be used for recognition dynamically at runtime based on the users current location. The features to be looked for were calculated from the textured model based on the current location and orientation. An Extended Kalman Filter was then applied for the frame-by-frame tracking. This system had many advantages over what had previously been done, the one of particular interest to this thesis being a reduction of storage

space. When performing object recognition in an outdoor scene the object must be analyzed from all potential positions to be reliable. With Reitmayr and Drummond's approach the required features were calculated on the fly. The original version of the algorithm was stationary, however the following year localization was added to allow the algorithm to be applicable from changing vantage points [35].

At this point in time, much of the work related to hybrid sensor augmented reality moved towards mobile phones as the platform of choice. Modern mobile phones come equipped with a camera, gyroscope, and GPS. From an availability standpoint, mobile phones are compact and have widespread usage, making them the ideal candidate for such work. Switching to mobile phones as the primary platform for OAR research brought new challenges alongside the added convenience. Weak camera, low accuracy sensors, and a dependency on having available network access being chief amongst them [2].

Once GPS functionality became standard for OAR in urban environments, it became mandatory to use localization to fine-tune the original placement in space. The high availability of smart phones to the average user allowed for the usage of crowd-powered localization algorithms.

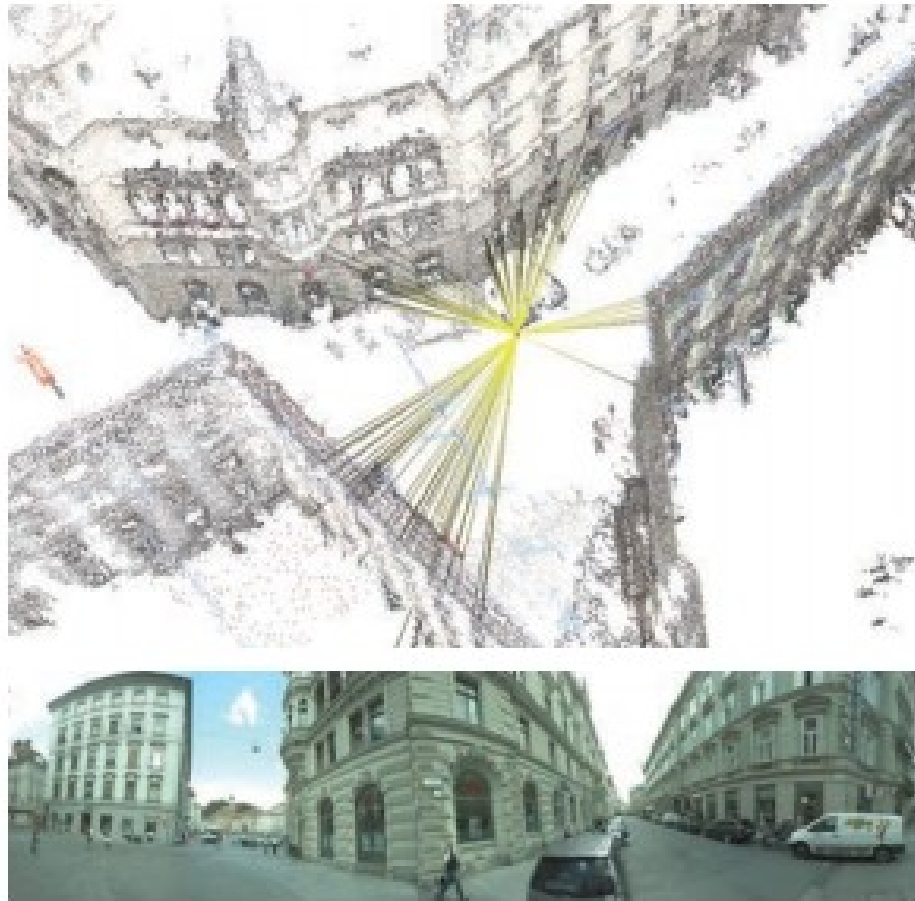


Figure 7: Localization from a panoramic image. Arth & Klopschitz [3]

This method was to have a tagged database of images from the region and find the set that best matched an image taken from the current location [42,36,3,8]. As for the method in which image collection would be to occur, a popular suggestion is to rely on the users to submit the geographically tagged photos [42,8]. Figure 7 shows an example of the approach by Klopschitz and Reitmayr, which performs localization from the vantage of a panoramic image taken of the scene [3]. In each of the listed approaches, image data is gathered by users with their cellular phones and a query is sent to a feature database (either local or external). When a match is found the server

will send back the list of feature points that it believes to be the most relevant, and the feature matching is performed. Whether this match is performed on device or at an external server is implementation specific.

In the late 2000s, two separate approaches became popular towards the mitigation of dynamic lighting that causes problem in regards to outdoor scene recognition. One focused on collecting and training over a large amount of data, the other on having less data and relying on world space assumptions [33].

The first approach was dense data collection. The premise behind this approach was to collect features for recognition and tracking concerning the target object in all conditions (including lighting conditions), and have faster (than the contemporary) algorithms to look up features found at the scene from their large feature database. This particular approach has a few drawbacks. A large amount of initial data was required for the system to achieve the requisite invariance to justify the usage of this approach. The second was that a large amount of data had to be stored and available for the user's current environment. In some cases [42], the feature look up and retrieval algorithms were performed by an external server (to compensate for a lack of processing power on mobile devices [2]) with the result of the look-up sent back to the user's device. This added the additional constraint of network availability.

The second approach (that on which the work presented here is based) sought to minimize the stored information per object. These algorithms were more reliant on

proper detection of preset patterns within the edge information as opposed individual feature look-up. Complex shape detection has strong resilience to the problems caused by dynamic lighting at the cost of more computationally intensive algorithms for the shape detection [2]. The second major disadvantage of this approach was that it was applicable in fewer environments than the dense data collection approach. Detecting shapes in a scene requires that such shapes exist in the current environment, therefore object recognition through pattern detection assumes that the pattern in question exist.

Buildings and other architectural structures benefit from the high regularity of their shape. When working with buildings this allows for a number of assumptions to be made about the scene reducing the amount of data required to be input into the system. In this regard there are multiple types of assumptions that are generally made when working with buildings.

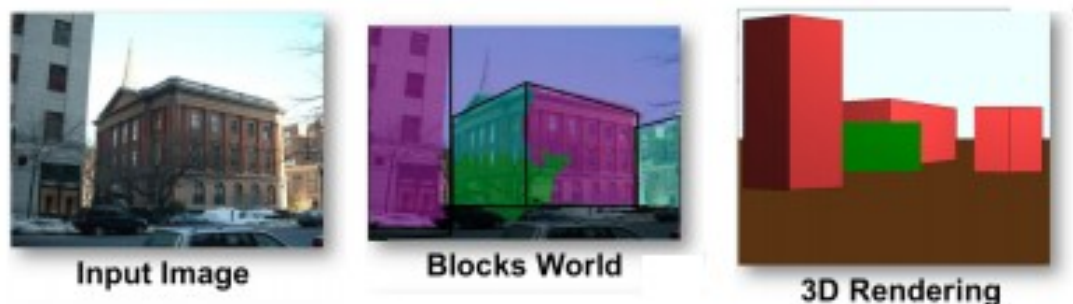


Figure 8: Creating a block drawing from an image. Gupta and Efros [12]

The first type of assumption is the block world assumption used by Gupta and Efros [12]. In this approach it is assumed that all buildings are made up of a combination of rectangular cuboids. The world is segmented and split as best it can be into regions that can be forced into rectangular cuboid regions. The regions are then matched with where they should be located in the scene according to world space physics assumptions. This method requires the ability to accurately segment the regions in the image space corresponding to the ground and sky in the world space. If the ground and sky have been properly segmented then the groups of edges in the image space with points of contact with the ground and/or sky go through a multiple method segmentation process before being initially categorized into their respective cuboids. The most interesting feature of this approach is how little input is required into the system. A strong assumption is forced onto the scene in an iterative fashion. Every object being searched for is a geometric “block” (rectangular cuboid). As such, groups of edges that do not confirm to block regions can be discarded. It is a powerful assumption with equally strong detriments.

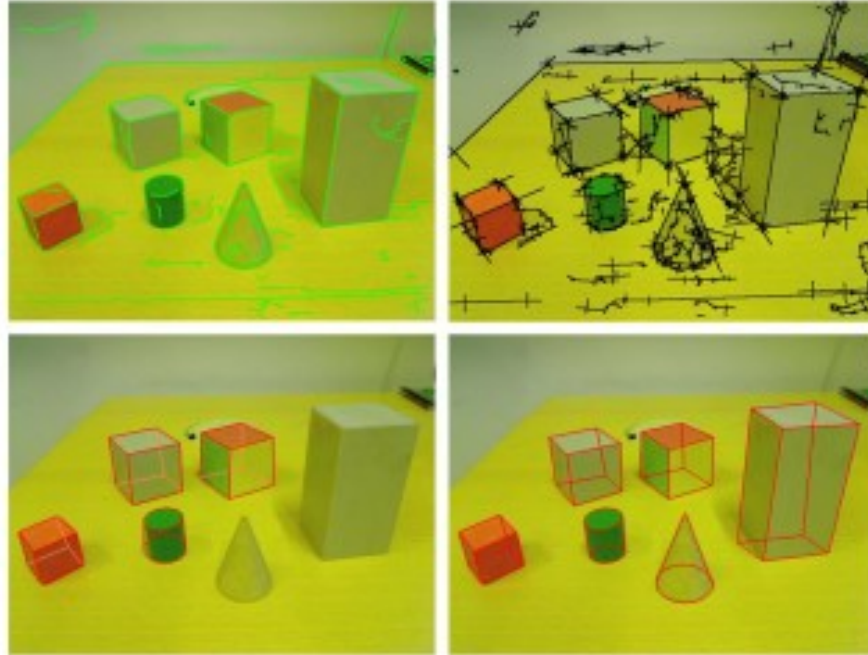


Figure 9: Detection of blocks placed on a table. Richtsfeld & Zillich [29]

Blockworld implementations suffer from issues with occlusion [29] or that the processing required for the block formation is too high to be used in a real-time application [12]. The second issue is not uncommon with model-based computer vision, particularly in the field of Augmented Reality. Model-based approaches most often perform the scene analysis (in particular the feature extraction) on the fly based on the user's position in the scene [33,2,50]. This aspect in particular makes them difficult to implement on mobile devices. Both the model and scene analysis can require a high amount of memory for storage, and mobile devices cannot be relied upon to have the processing power to perform the analysis quickly [2].

The second type of assumption is planarity. The assumption in this case is that buildings in the world space are made of a set of connected planes (the walls). In

some cases it is worthwhile to make the stronger assumption that each plane has a series of line segments that go either vertically towards infinity in the y-axis or horizontally towards a vanishing point in the world space [41]. Figure 10 provides an example of a building in the image space where the line segments of the building of interest are color coded based on their believed vanishing point.



Figure 10: Vanishing Point grouped line segments. Stamos, I., & Allen 2000 [41]

Ventura & Höllerer [8] provide information on both the advantages and limitations of this approach. Storage space is minimized, as generally only planar and location information need to be stored for the object in question. A way to consistently detect the planarity of pixels is required for this method to be successful, and to do so reliably requires extra sensors (some form of accurate range finder) in addition to the

standard gyroscope and GPS. When performing this plane-based approach without a range finder, a non-insignificant amount of user interaction is required for localization [10]. New Structure From Motion (SFM) work being done on the detection of symmetric structures in outdoor scenes could potentially alleviate the reliance of user interaction for plane detection [7].

In their implementation, Stamos and Allen used a range finder for the detection and confirmation of planes. Lines and pixels are clustered based on the plane to which they belong and this information is stored as the model for the building. In this work Stamos and Allen also mention the difficulty in accurately moving from 2D images to 3D planes without either 3D depth information or strong assumptions about the scene [41].

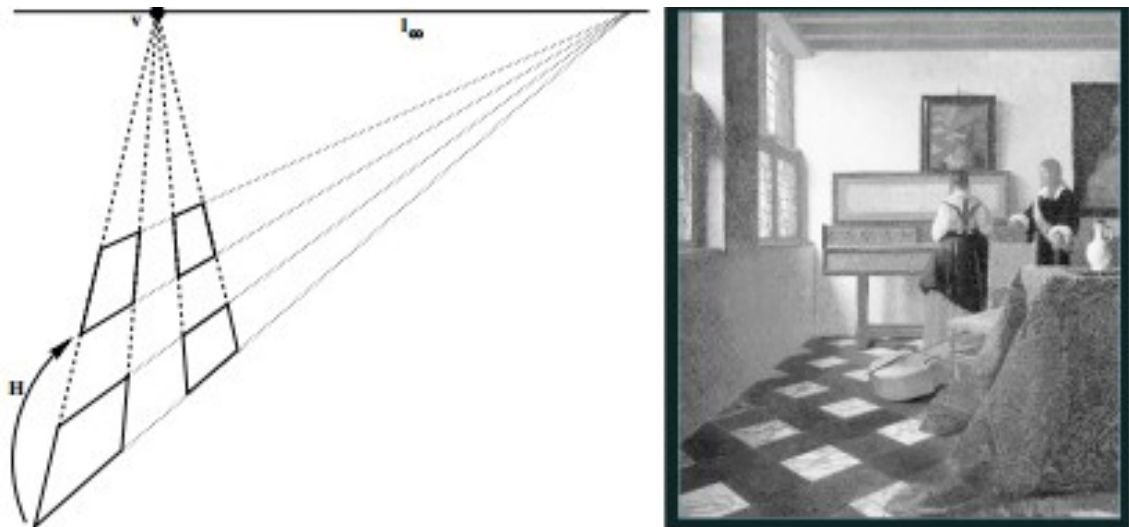


Figure 11: Projection of a plane and its vanishing points from world-space onto the image space. Schaffalitzky and Zisserman [39]

Assumptions made about the location of suspected vanishing points in a scene could provide much of the necessary information. In 1999 Schaffalitzky and

Zisserman explored using vanishing points to link planes from the 3-dimensional world space to a 2-dimensional image space [39]. Schaffalitzky and Zisserman's approach grouped lines found in the image space based on a set of conditions (such as regularity of spacing, estimated regions of intersection, etc) in relation to a point believed to be the vanishing point. The grouped lines could then be considered to be the projection of a plane in the 3-dimensional world space onto the image space.

CHAPTER 3

SIGNIFICANCE OF STUDY

Augmented Reality (AR) is currently in a state of crowd-sourced development. Developers and Users participate in the creation of mobile AR apps and simulations. For an indoor scene AR users are able to create AR scenes with little knowledge of vision or, more specifically, image recognition. Users and developers can take advantage of popularly available toolkits and frameworks such as PTAM [17] to handle the complexities of tracking and mapping. Outdoor AR on the other hand does not allow for simple usage.

In 2012 Takeuchi and Perlin presented work on the Elastic City [43], demonstrating the ability to augment and modify an outdoor environment with basic computer vision techniques. The focus on the work was the idea that Outdoor Augmented Reality had entered a state where non-vision specialized developers have the ability to develop OAR applications. Unfortunately, collecting the prerequisite information was still a bottleneck, limiting to OAR small environments and only to those with the capabilities to gather detailed information about the scene. The statement of this thesis is that with proper model selection for the environment, it is possible to perform outdoor object recognition with little initial configuration. The prototype algorithm constructed to demonstrate this is the focus of this work.

The world model for this approach is similar to that of a block world. The assumption is made that all buildings in the world are made of rectangular cuboids. The limiting factors of this world assumption is that environments with buildings that can not be represented as rectangular cuboids cannot take advantage of the proposed system. The benefit gained from this assumption however, is that all buildings are uniquely identifiable based on their location and the relation between their four walls (the four sides/planes of the cuboid). In this work the walls of buildings are considered planes identifiable by the grid formed from their windows and doors. This makes each building a set of four grids.

Viewing buildings as planes of grids formed by the windows is not new. In fact it is currently a popular way to store models of buildings to reduce the size and complexity of storage [34,49], increase scene understanding [5,49], for the generation and modeling of buildings in virtual environments [18,34], and building recognition [44]. An example segmentation of the face of a building into a grid from [34] is shown below in Figure 12.

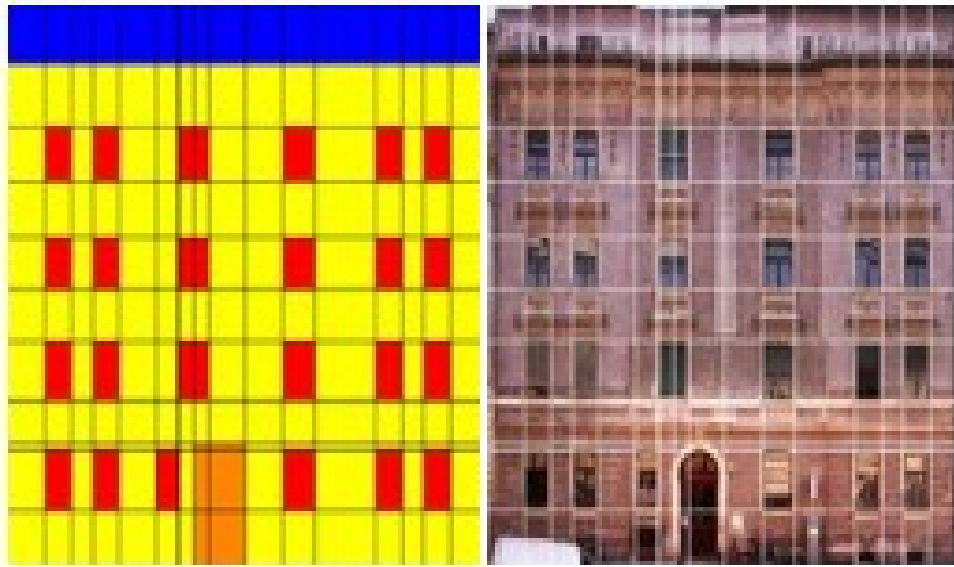


Figure 12: Building Grid Layout. Riemenschneider, Krispel, and Thaller [34]

METHODOLOGY

The presented prototype of the proposed approach is organized into two components. The first component, the recognition component, is that which is detailed below. The second component, the tracking component, is something that will be constructed and assessed in future upcoming work.

The recognition component identifies buildings in a four-step process. The image is first processed into a set of line segments. Vanishing points are then located in the image from the line segments. Planes potentially corresponding to the faces for buildings are then located in the image based on the vanishing points. Finally, the planes extracted from the image are matched against the known faces for buildings.

Basis of Work

The recognition component identifies and matches buildings based on their faces. The plane detection and recognition used in the below presented algorithm is an implementation of the plane identification algorithm used by Trinh and Jo. Additionally, the template-based plane recognition method for building recognition is most similar to the algorithm by Johansson and Cipolla [14], differing only in terms of the rectification method used and the assumptions regarding objects in the world. Of special note is that the work of Johansson and Cipolla was not extensively explored due to the high performance costs required of their template matching algorithm [33]. The algorithm presented in this work places assumptions on the geometric shape of the

target object in order to reduce the search space, while the approach by Johansson and Cipolla allows for their system to be used on a much broader variety set of target objects.

Overview

The core of the recognition phase is the identification of buildings in a scene based on their vanishing points. As can be seen from the system overview diagram below, the system begins with the input of the source image to the image pre-processing (IPP) component. The lines extracted during the IPP component are sent to the vanishing point detection component. The list of detected vanishing points are sent to the plane detection component. The objects in the image believed to correspond to planes are sent to the plane matching component, which compares them to planes stored in the model. The plane matching component then outputs a score representative of how strong of a match the two planes have with each other.

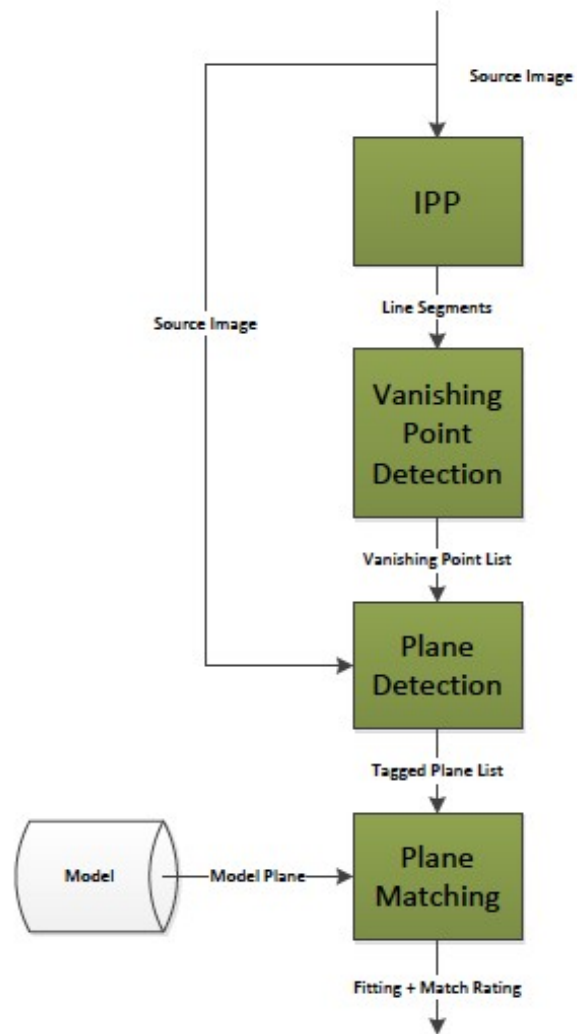


Figure 13: System overview diagram

Image Pre-Processing:

The first component of the recognition system is the pre-processing. The goal of this component (shown in Figure 14) is to perform low-level image processing and group the line segments by their suspected vanishing points.

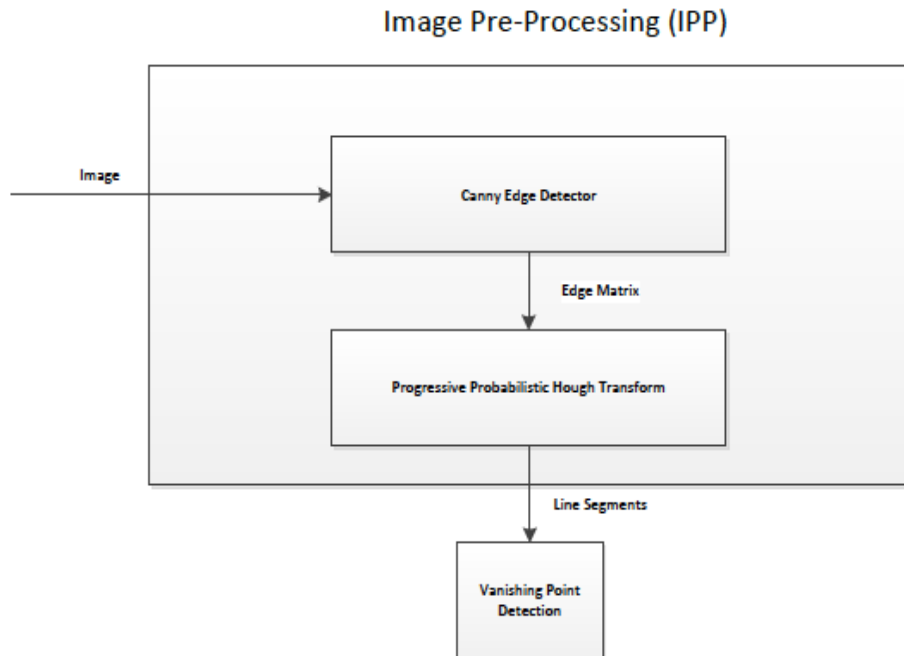


Figure 14: Image pre-processing

When given an image as the input to the system, the Canny edge detector [6] was chosen for the initial segmentation of edge pixels. While the processing required for the Canny detector is greater than for other comparable edge detectors, it has the lowest misclassification rate as long as the initial parameters are correctly set [21]. The output from the canny detector is then input into the Progressive Probabilistic Hough Transform (PPHT) [27] which returns a list of line segments. The PPHT

algorithm has a low processing time making it optimal for real-time systems. Additionally it reliably finds line segments with minimal pre-configuration [27]. The list of line segments returned from the PPHT algorithm is then sent to the vanishing point detection component.

Vanishing Point Detection:

During the vanishing point detection process, the line segments are inspected in terms of their relationship to their believed vanishing point. When the world is projected into an image, the lines that were parallel in the world may no longer be parallel in the image. The line segments in the image corresponding to these formerly parallel lines in the world will converge at a singular point known as the vanishing point. All lines parallel to each other in the world will converge at the same vanishing point in the image. Most importantly to the presented algorithm, all parallel lines belonging to the same plane will converge at the same vanishing point.

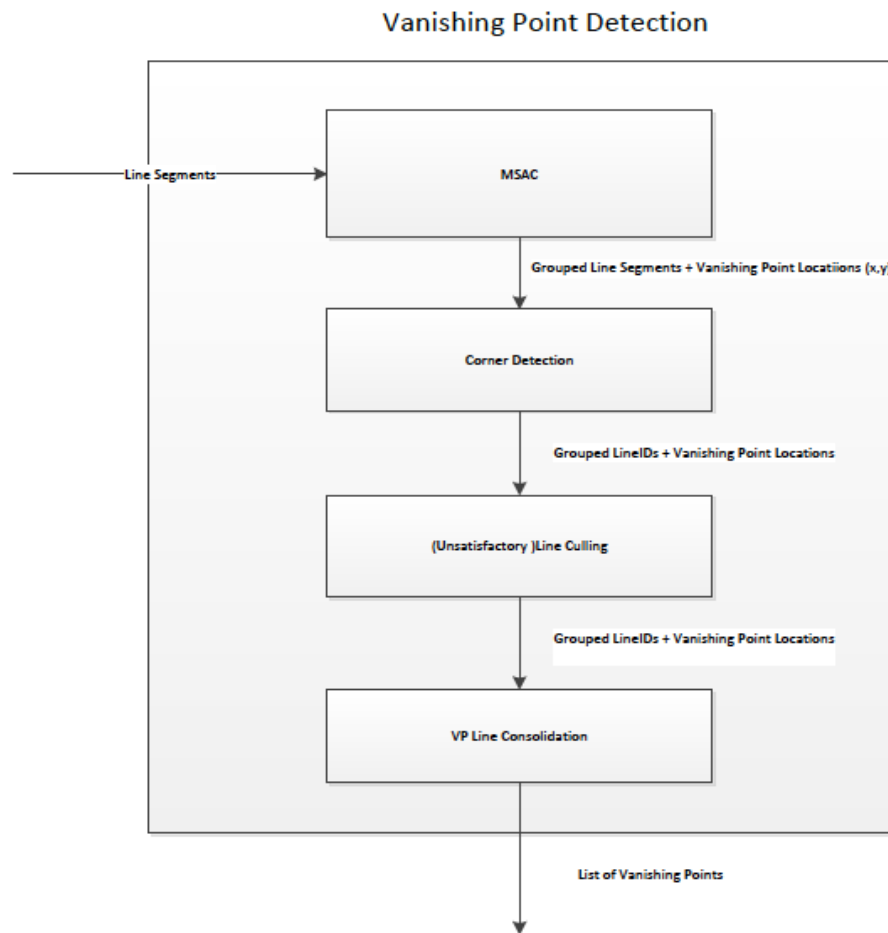


Figure 15: Vanishing Point Detection.

The Vanishing Point Detection component (shown in Figure 15) begins with the list of line segments given from the PPHT. This list of line segments is then provided to the MSAC algorithm [30]. The MSAC algorithm will estimate the location of vanishing points in the scene and cluster the provided line segments based on which vanishing point to which they belong. The list of line segments as well as the locations of their respective vanishing points is sent into the corner detection algorithm.

In an outdoor scene, the image data received will suffer from issues of both occlusion and lighting. For this reason, Zillich and Vincze's algorithm [51] for detecting closed polygonal regions from edge lists was used as the corner detection algorithm. The basis of Zillich and Vincze's algorithm is a greedy method for extending and connecting nearby edges (line segments) that provides strong resilience against missing edge information in an image. The drawback of this method is that artificially extending line segments risks adding connections where none previously existed. However, for this implementation, that is not a worry. The output of this algorithm is the list of lines organized by their vanishing points. At this stage each line is now stored as a Line Identifier (LineID) object that stores the location of the original line segment, its six extension points as returned via the corner detection algorithm, a list of the line's intersections with other lines, and well as a reference to the location of the vanishing point to which the line belongs.

For each vanishing point that was detected a Vanishing Point object is now created. The Vanishing Point object stores the location of the vanishing point, as well as the list of Lines that belong to the Vanishing Point. Lines are then consolidated based on their polar coordinate using their vanishing point as the origin. Lines belonging to the same vanishing point are consolidated if their rho and theta values are in too close proximity. This was done to ensure line segments belonging to the same line (duplicates) and lines that are indistinguishably close are not counted multiple times. After removing any close and repeated lines the vanishing points are sent to the planar detection component.

Planar Detection:

The Planar Detection component (shown in Figure 16) checks if any of the lines belonging to the vanishing points from the vanishing point detection component could correspond to the projection of a plane from the world space. The plane detection and matching components are based on Zisserman's work on planar rectification [20] and vanishing point detection [39].

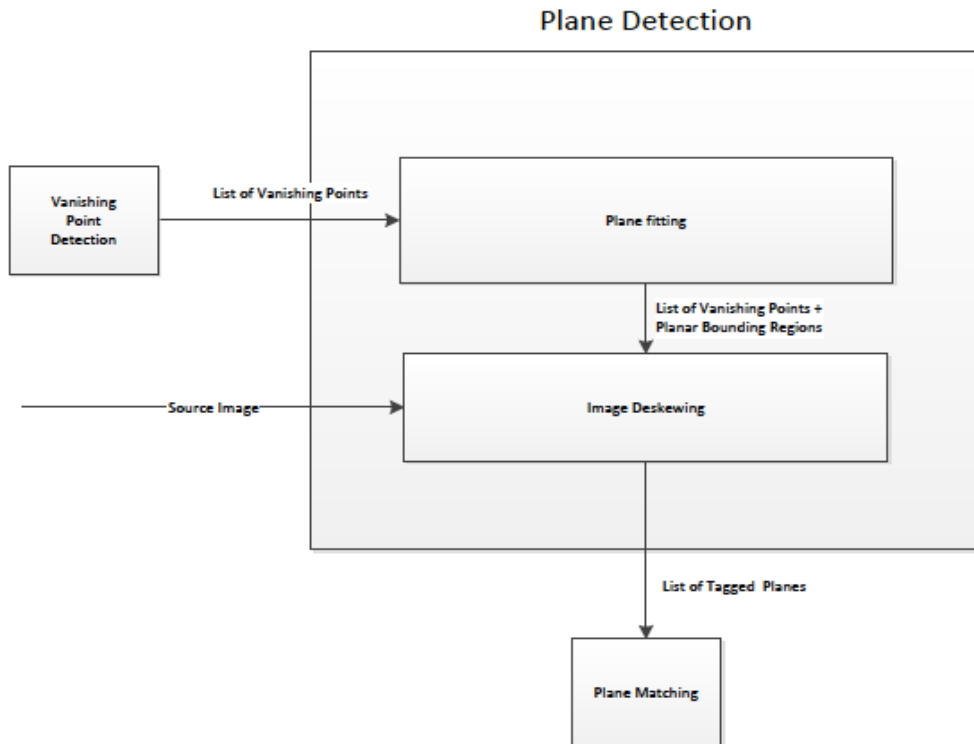


Figure 16: Plane Detection

The assumptions from this section are based on the planar work of Schaffalitzky & Zisserman [39], in particular that a plane in 3-dimensional world space, when represented in 2-dimensional image space, will correspond to a transformed grid

where the lines of the grid belong to one of two vanishing points. For architectural structures, where the faces of the structure consist of vertical planes, one of the two vanishing points is located at infinity of the z-axis in 3-dimensional world space as seen in Figure 11.

For this initial version an assumption has been made that the pitch is level. The advantage of this assumption is that the vanishing point located at infinity in the z-axis of world space, now to be referred to as the vertical vanishing point (VVP), is mapped at infinity of the y-axis in the 2-dimensional image space. All suspected vanishing points found in the scene that are not the vertical vanishing point, will be referred to as horizontal vanishing points (HVP).

At the beginning of the planar detection component, an attempt is made to bound a quadrilateral region onto the list of segments associated with each horizontal vanishing point, based on the intersections between line segments belonging to the horizontal vanishing point and the line segments belonging to the vertical vanishing point. Figure 17 shows on the left a plane with the horizontal and vertical vanishing lines overlaid on top. The two corner intersections that provide the largest bounding size are selected by iterating through the intersection list of all lines belonging to the horizontal vanishing point and testing if using the corner corresponding to that intersection would increase the area of the quadrilateral.

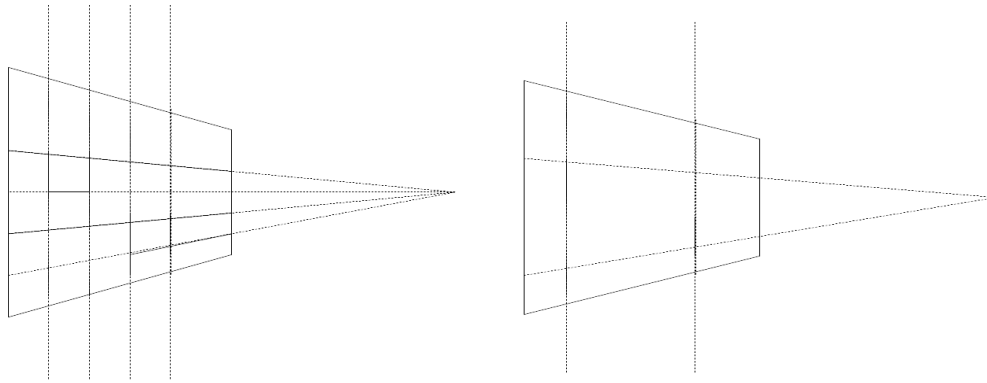


Figure 17 Left:: Vertical and Horizontal vanishing lines crossing over a plane in the image; Right: The vanishing lines providing the largest area quadrilateral

Once the two corners are selected a quadrilateral is formed from the four intersection points between the upper and lowermost line belonging to the horizontal vanishing point that pass through the two selected corner points, and the left and rightmost lines belonging to the vertical vanishing point that pass through the two selected corner points. Figure 17 shows on the right the vanishing lines with the largest bounding quadrilateral. Each vanishing point that is able to be bound with a quadrilateral is considered to contain a plane (Figure 18 displays a labeled planar region in an image). Once the four bounding lines are selected, the quadrilateral bound by the four vanishing lines is rectified into a rectangular grid.

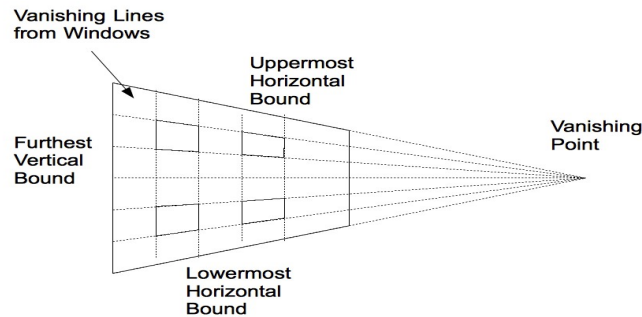


Figure 18: The quadrilateral projection of a face of a building.

The biggest benefit of storing the planes as rectangular grids is storage. A grid consists of only two lists, one for the lines along the x-axis and one for the lines along the y-axis. To rectify the quadrilateral region into a grid three things must be done. The first is that the bounding quadrilateral must be transformed into a rectangular region (shown in Figure 19). To do this, the horizontal (upper and lower) bounding lines for the quadrilateral must be rotated to x-infinity. For this rotation the intersection between each line and the vertical bounding line (leftmost or rightmost) furthest from the vanishing point is used as the origin of the rotation. One important thing to note for this rotation is that without depth information the rotation will lack the information needed to rotate along the z-axis, warping the dimensions of the output region.

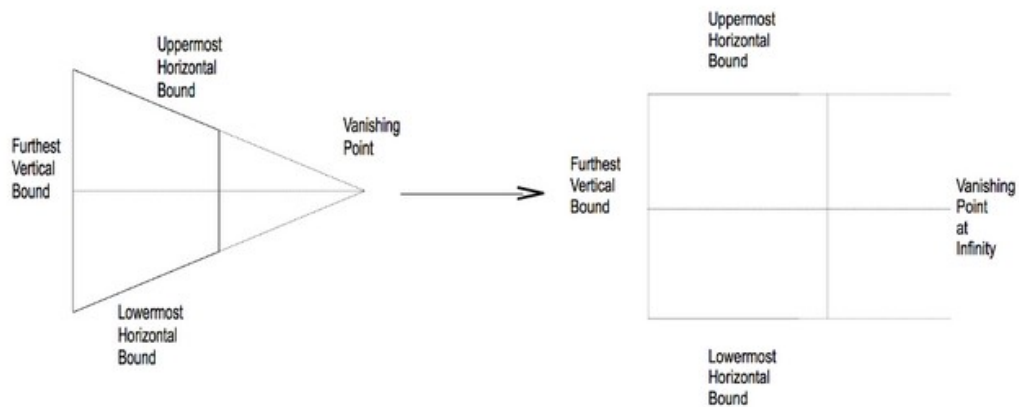


Figure 19: Rectified bounding quadrilateral

The second step required for rectifying the region is to rotate the lines belonging to the horizontal vanishing point to x-infinity. A line in the direction of the x-infinity vanishing point has constant y-value. As such, the horizontal vanishing line can be described by this y-value. As this y-value is constant, rotating the line is not

necessary. Given the intersection of this line with the vertical bounding line that is furthest from the horizontal vanishing point, the y-value of this point is the characteristic y-value. For each line belonging to the horizontal vanishing point store this value in a list of horizontal points.

In order to find the characteristic x-value of the lines belonging to the vertical vanishing point (location at y-infinity) that exist within the bound region, the lines must be rotated. For each line belonging to the vertical vanishing point in the bound region, the intersection of the line with the upper or lower horizontal bounding line must be found. This point is then rotated towards x-infinity using the intersection of the horizontal bounding line and the furthest vertical bounding line as the origin of rotation. The x-value of this rotated point is the characteristic x-value for the line. The rotation process for the vertical lines is shown below in Figure 20, where point M is the origin of rotation.

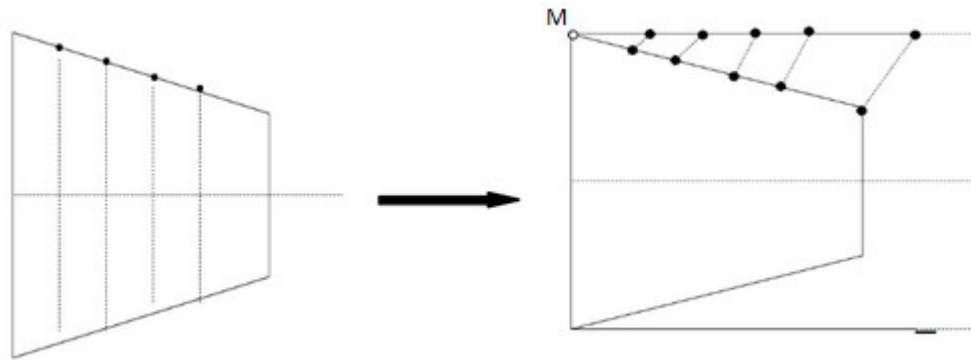


Figure 20: Rotation of vertical lines towards x-infinity

The final step of the rectification process is to take the region of the image corresponding to the bound quad and transform it to fit in the rotated bounding rectangle. The pixels should be rotated with respect to the vertical bounding line furthest from the horizontal vanishing point. Interpolation will need to be performed over the pixels for the gaps in the bounding rectangle (the bounding rectangle is larger than the bounding quad, as such there will be gaps in the rectangle after all pixels have been rotated and placed). This implementation used a nearest neighbor approach for the pixel interpolation. Nearest neighbor interpolation is equivalent to setting the gaps to a weighted average of their neighbors. This specific interpolation algorithm was chosen because of its ease of implementation. Once these three things are done, a tagged plane object is created to store the vertical and horizontal line lists that represent the grid of the plane, and the rectified image region (the pixel map). The tagged planes are then sent to the plane matching component.

Storage Model:

At this point it is necessary to describe how objects to be found in the scene are saved in the model. An assumption that has been made for this initial release is that all buildings are cuboid in shape (as shown in Figure 21).

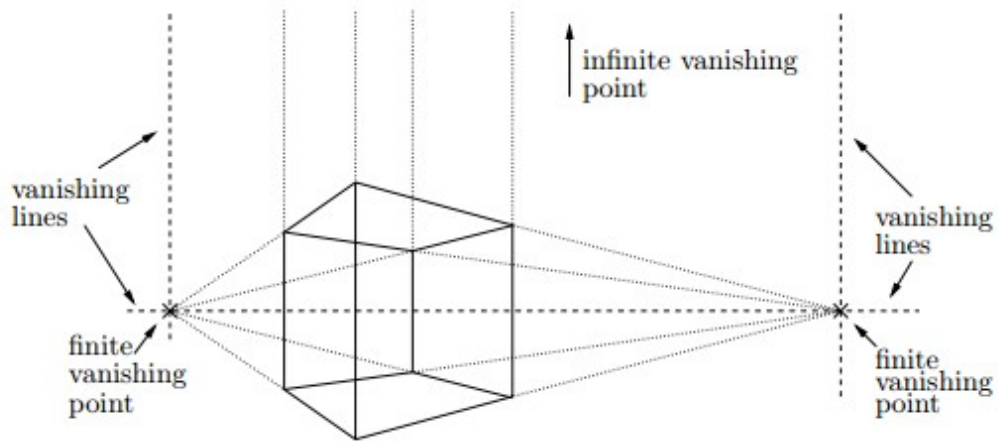


Figure 21: A cuboid model. Rother 2002 [37]

A building is stored as a list of its connected planes. Each plane has a list of the locations for each vertical and horizontal vanishing line if in the 2-dimensional image space the vertical vanishing point is at y -infinity and the horizontal vanishing point at x -infinity (fronto-parallel to the camera).

Each plane is also stored with a set of features computed from the grid formed by its x and y lists. A sample of the texture for the plane is stored as a hue histogram. Hue is stored as it is invariant to changes in white light levels. The cells of the grid containing windows (or other sections with non-static textures) on the plane must also be marked and stored. The location of these cells must be known, as they are detrimental in a similarity scoring of the texture for the plane. The texture of windows varies depending on time of day, distance, content behind the window, as well as any

other number of unknown criteria, as such recording the location allows for the ability to avoid incorporating these unknown regions when a texture comparison between the model and the scene is performed. The feature point used are the ratios of the diagonals between corners of the grid. Using these diagonal features has two major advantages. The first is that this feature is scale invariant. The second is that the feature is not unique. When working with architectural constructs, symmetry along either the x or y axis is often an assumption that can be made [11]. As such repeated feature values are tossed, reducing the storage space.

Equally, there are also two major disadvantages to using diagonal features. These disadvantages affect the feature extracting and matching component of the algorithm. The first disadvantage is that unless the algorithm possesses accurate localization to properly estimate how much of the plane found in the scene has been lost in the z-axis, unless the plane's horizontal vanishing point is located at x-infinity in the image space, the loss of depth information will result in incorrect features being extracted from the scene. The second disadvantage is the lack of uniqueness in the features. This increases the number of potential poses for the model that must be tested in the feature matching component, as each feature can exist at multiple locations.

Plane Matching:

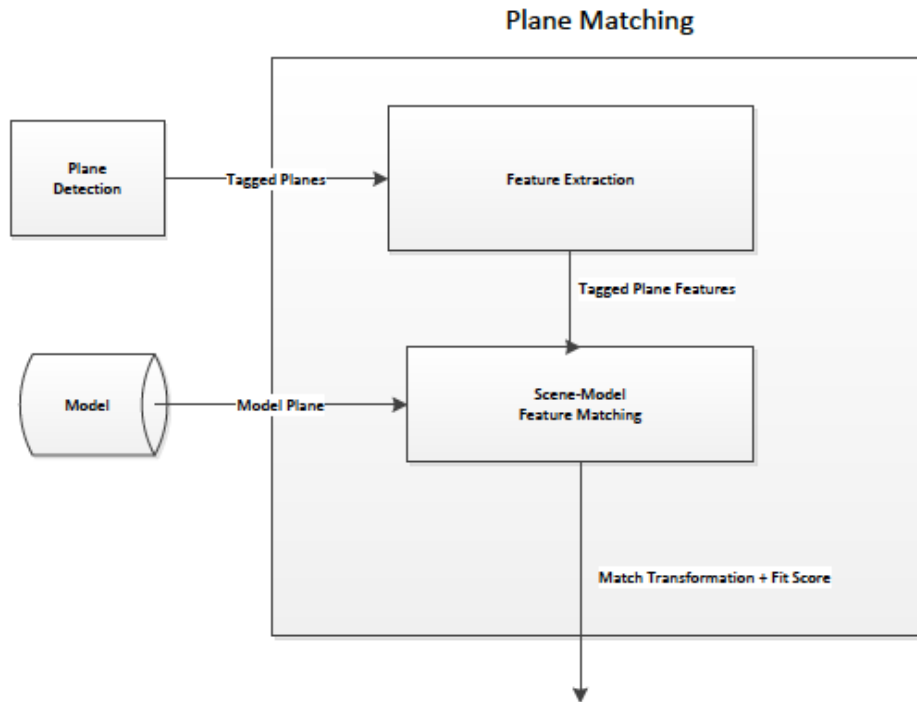


Figure 22: The Plane Matching Component

A potential plane brought as input from the Planar Analysis component (Figure 22) has its diagonal features extracted from its grid. These features are then matched against the features of each plane in the model. Matching the features will provide a guess to the scaling and position of the plane from the scene against the plane in the model. For all scaling and positions that score above a certain threshold the planes go to the histogram comparison step. For the histogram comparison step the potential plane from the scene has a hue histogram calculated based on what are the non-windowed cells of its rectified grid for the given scaling and position. This newly calculated

histogram is compared with the hue histogram stored with the plane from the model using a correlation matrix. The biggest advantage of using a correlation matrix for the histogram comparison is that a correlation matrix normalizes the size of the histograms. As the sizing of the plane from the scene and the plane from the model are independent from each other, it is necessary for the histogram comparison algorithm to be scale invariant.

CHAPTER 4

RESULTS

System Specifications

The development environment for the system was as follows:

Operating System: *Mac OSX 10.6*

CPU: *2.4 GHZ Intel Core 2 Duo*

Memory: *4 GB DDR3 RAM*

The target environment of the developed system would be a mobile device or tablet.

Once satisfactory performance is achieved with the current development environment an effort will be made to port the system to the expected user environment.

Component Testing

In order to measure the functionality of the presented system, three of the base components (the vanishing point detection, planar rectification, planar matching) were tested. Quantitative testing was performed over purely simulated data to measure the performance of the vanishing point detection component. Afterwards, testing was performed on the vanishing point detection, planar rectification, and planar matching components over a generated model of a virtual environment to illustrate the findings and effects of the quantitative testing.

Test Resources

The images used for the tests were constructed using the computer modeling application Blender. Blender was chosen both because it is open-source and due to the author of this study's previous experience using the application. The model used to generate the image was used under a Creative Commons Zero license [1].

Model Testing

The components were tested on three images taken of the same sample computer model from differing vantage points. The first (Figure A) is the base case, where the target plane is near fronto-planar by default and close to the camera. The second image (figure B) is a rotation test, where the object of interest has undergone a strong rotation in regards to camera. The third image (figure C) is a distance test, where the object of interest is far from the camera.

Vanishing Point Detection

The vanishing point detection component is concerned with quickly estimating the location of each vanishing point and organizing the line segments based on which vanishing point to which they belong. The algorithm will greedily assign line segments to vanishing points as they are found, in a way that minimizes the number of vanishing points detected. As a consequence, the line segments are under fit and often misclassified to vanishing points to which they do not belong. In the following three test images, the green segments belong to the vertical vanishing point, and the blue segments to the primary horizontal vanishing point. The black quadrilateral is the bounding quadrilateral detected in the image.

Quantitative Test: Vanishing Point Detection

Test Conditions

The vanishing point detection component was tested over a simulated object with two planar faces, each face containing 100 equally spaced line segments, 50 vertical and 50 horizontal. The simulated object was rotated away from the camera by a constant 60 degrees during the testing. The ordering for the list of line segments belonging to both faces was randomized and classified by the MSAC algorithm 200 times.

Test Overview

This is a robustness test of the Vanishing Point Detection component for a generated object. The generated line segments were classified by their relationship to their true vanishing point and a measurement was taken of the success rate for their classification. Three vanishing points existed in the generated data.

Measured Values

- Average success rate of line classification: 70%
- Maximum success rate over a single test session: 75%
- Minimum success rate over a single test session: 50%

Results

The line segments from this step would be used to compute a bounding quadrilateral for rectifying planar faces found in the scene. The quadrilateral bounding (and thusly the rectification) relies on the line segments being accurately grouped by the correct vanishing points. Currently, the quadrilateral bounding calculations possess the possibility to fail due to the misclassification of a single line segment (as will be seen in a later test case). As such, the vanishing point detection component does not meet the required rate of stability.

Model Test: Vanishing Point - Baseline Test

Test Conditions:

The line segments in the image belong only to the vertical vanishing point or a single horizontal vanishing point. The horizontal vanishing point is located at x-infinity.

Test Overview:

This is the optimal condition test. Only two vanishing points exist in the image. Additionally, the two vanishing points the maximum difference in angle between them (90 degrees). If the algorithm were to fail this case it could mean an irreconcilable problem in the vanishing point estimation.

Results:

As can be seen from the generated image below (Figure 23), the quadrilateral with the highest area from the corners found was detected. Additionally, all detected edges were correctly classified as belonging to the appropriate vanishing point

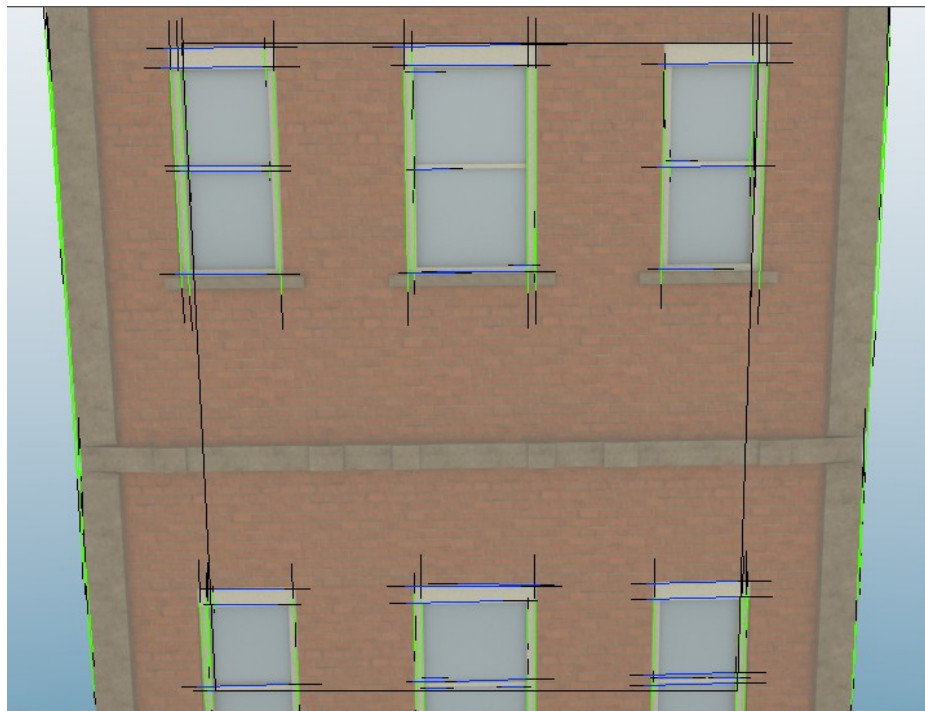


Figure 23: Baseline line segmentation and bounding

Model Test: Vanishing Point – Rotation Test

Test Conditions

The target object has been rotated away from the camera. There are only two vanishing points in the image. The horizontal vanishing point is not at x-infinity.

Test Overview

This is the rotation test. Only two vanishing points should exist in the image. However, due to the low threshold set on the Hough Probabilistic transform for line detection, a few phantom diagonal lines were detected in the image. Only a few of these lines existed, as such they should be marked as extraneous by the MSAC algorithm (or subsequent line confirmation) and ignored.

Results:

None of the line segments that exist in the image below are mislabeled (Figure 24). However the phantom line segments (lines added in because of low thresholding on the Hough Transform), were not discarded and instead assigned to the vertical vanishing point. This means that more stringent confirmation of the vanishing point to which line segments belong is necessary.

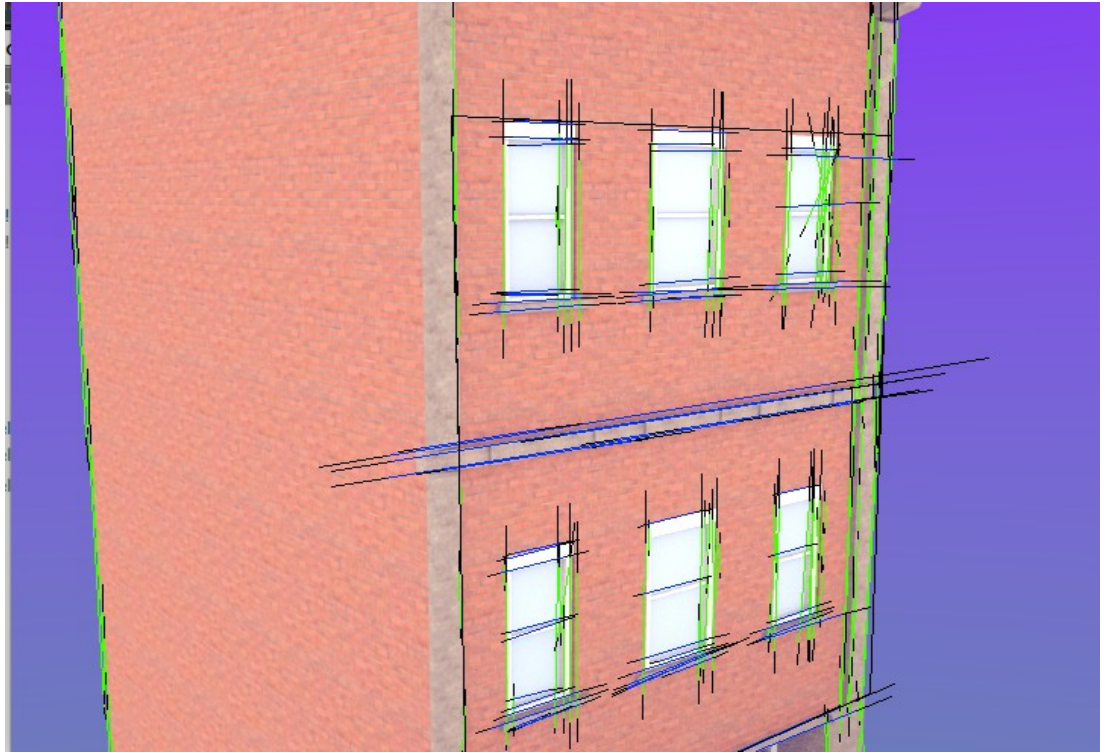


Figure 24: Rotated line segmentation and bounding

Model Test: Vanishing Point - Distance Test

Test Conditions:

The camera is further from the target object than in the optimal case. An object not belonging to the target object (the road) is in the scene. For this case there are line segments in the image that do not belong to the two primary vanishing points. These are the two corner lines of the target building corresponding to the face of the building adjacent to the target face.

Test Overview:

This test has two suboptimal conditions. The first being the existence of lines belonging to a third vanishing point. As there are only two line segments that belong to this vanishing point, it would be acceptable behavior for the algorithm to either ignore the two lines or group them as belonging to a third vanishing point. The second condition is that there is a second object in the image that shares a vanishing point with the plane of interest.

Results:

As seen in Figure 25 below, the line segments that do not belong to the two primary vanishing points have been misclassified as belonging to the vertical vanishing point. This causes an error in the creation of the quadrilateral bounding, demonstrating the worst error case, where the misclassification of a single line segment has led to an unusable quadrilateral bounding. Additionally, one of the line segments belonging to the sidewalk was selected for the lower bound of the building's face due to its sharing a horizontal vanishing point with the plane of the building. If the line segments belonging to the third vanishing point were not detected as belonging to a third

vanishing point, they should have been discarded during the line segment confirmation. The merging of multiple surfaces sharing a vanishing point is a problem that occurs with the approach used. However it is something that should be accounted for with future work on the building recognition component and is out of the scope of this current work.

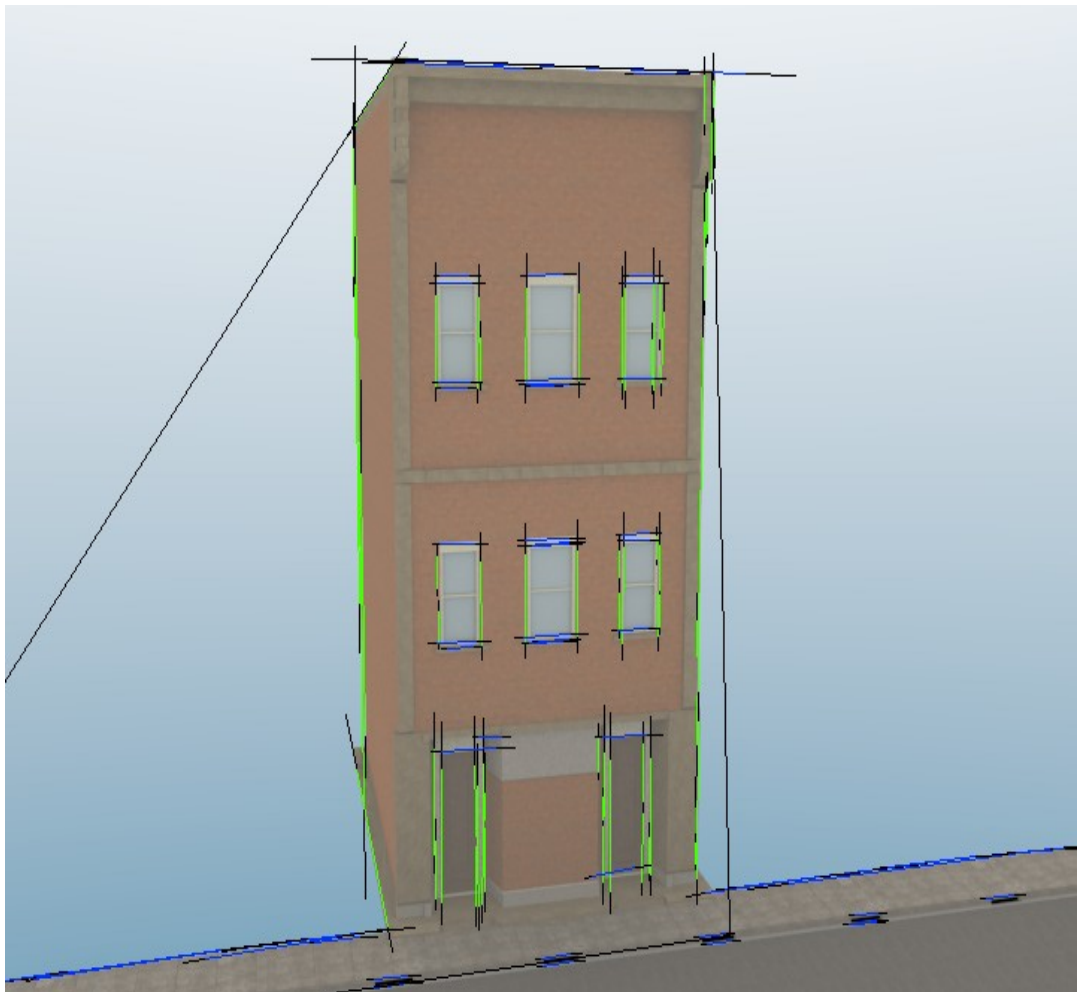


Figure 25: Distance line segmentation and bounding

Planar rectification

The planar rectification requires having an accurate quadrilateral bounding box from the vanishing point detection step. For this reason, only images A and B have been used for testing the planar rectification (explanation provided in test, "Vanishing Point – Distance Test"). For the accuracy metric, the measurements between the corners of the rectified planes will be compared to the measurements of the manually entered template image (Figure 26) guiding the recognition of the planar surface. For the provided template the vertical lines are indexed by letters and the horizontal lines by numbers. The blue squares correspond the windowed regions in the plane.

	A	B	C	D	E	F	G	H
1								
2								
3								
4								
5								
6								

Figure 26: Plane template (windows in blue)

Model Test: Planar Rectification - Baseline

Test Conditions:

This is the baseline case. The lines belonging to the vertical vanishing line are vertical towards y-infinity in the image plane and the lines belonging to the horizontal vanishing line are horizontal towards x-infinity in the image plane.

Test Overview:

In this case, line rotation is not required of the rectification algorithm. The region provided from the quadrilateral detected in the scene was already front-planar. The algorithm must detect that the line segments are within admissible bounds and not rotate them or rotate them to such a small degree that the grid is preserved. The region encapsulated by the given quadrilateral contained all three of the upper windows and half of the lower windows. Therefore it had width from regions (B-G) and height from (1 – 4.5). The width and height ratios between many of the key corners in the template were compared to their equivalent measurements found by the algorithm to judge the accuracy of the line rectification. The first set of measurements to compare are the ratios computed for the individual cells between the two grid. Afterward, the largest length width corner ratio is compared.

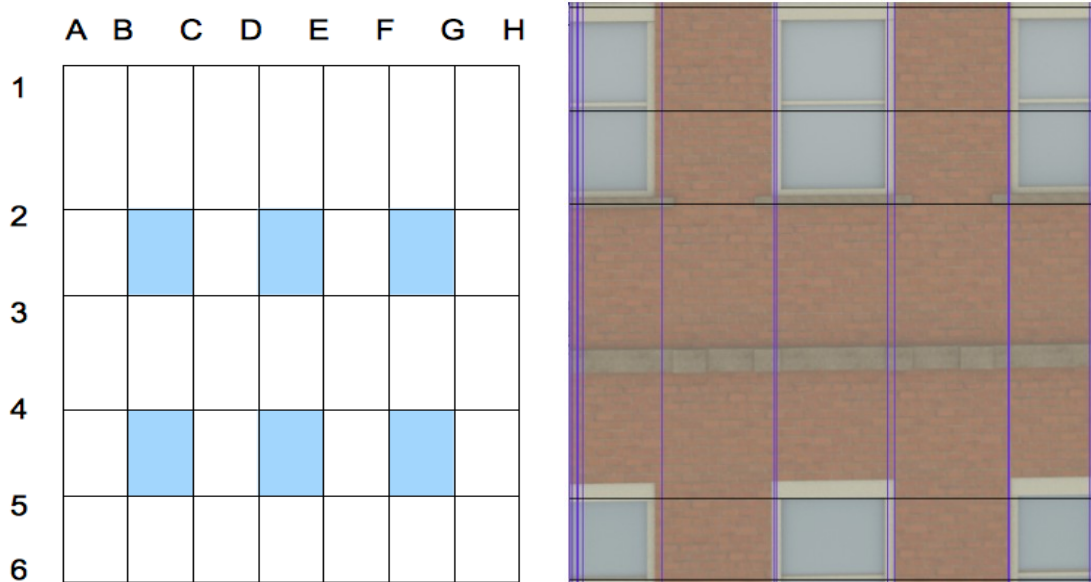


Figure 27 Left: Labeled template; Right: Grid segmented baseline image

Results

For this test there was a perfect matching between the cells in the template and the cell in the baseline image (Shown in Figures 27a and 28b respectively). This is not surprising as the baseline image requires minimal (if at all) rectification and rotation. This demonstrates that when rectification is not required, the algorithm is capable of finding and correctly parsing the plane found in the scene into a grid. The comparison of the cell-wise measurements between the template and detected plane are below in Figures 28 and 29. The width comparison between the template and detected plane are shown in Figures 30 and 31.

Cell Comparison

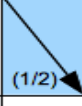

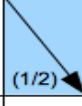

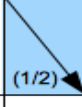
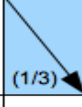
	A	B	C	D	E	F	G	H
1								
2								
3								
4								
5								
6								

Figure 28: Template with cell-wise measurements

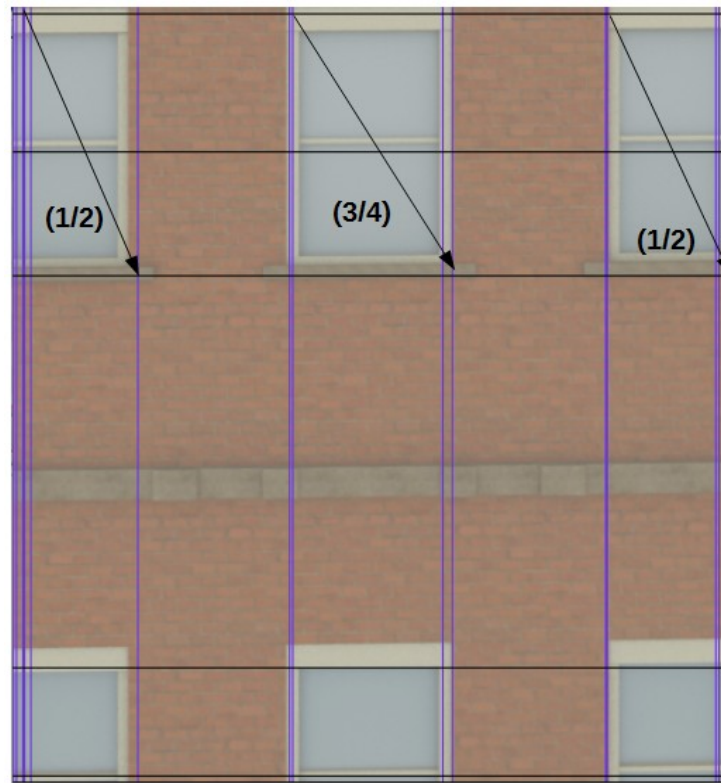


Figure 29: Grid segmented baseline image with cell-wise measurements

Width Comparison

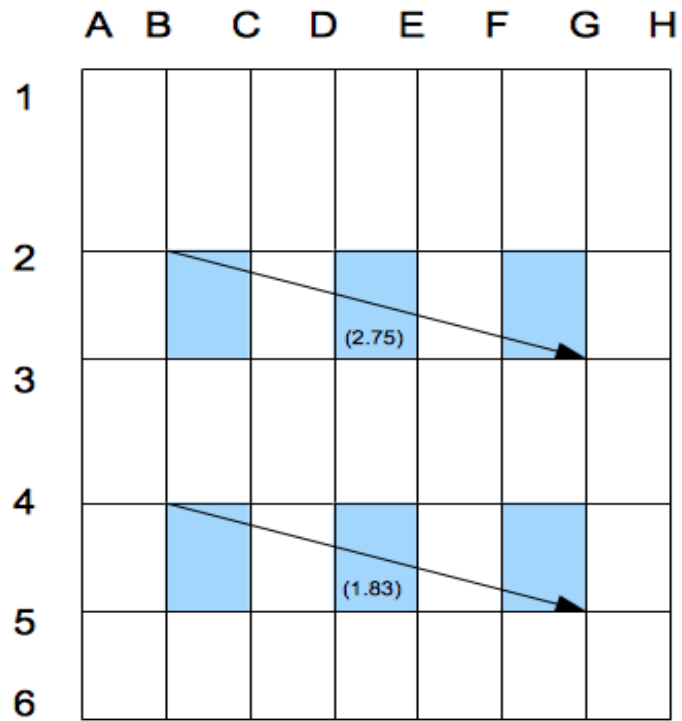


Figure 30: Template with width measurements

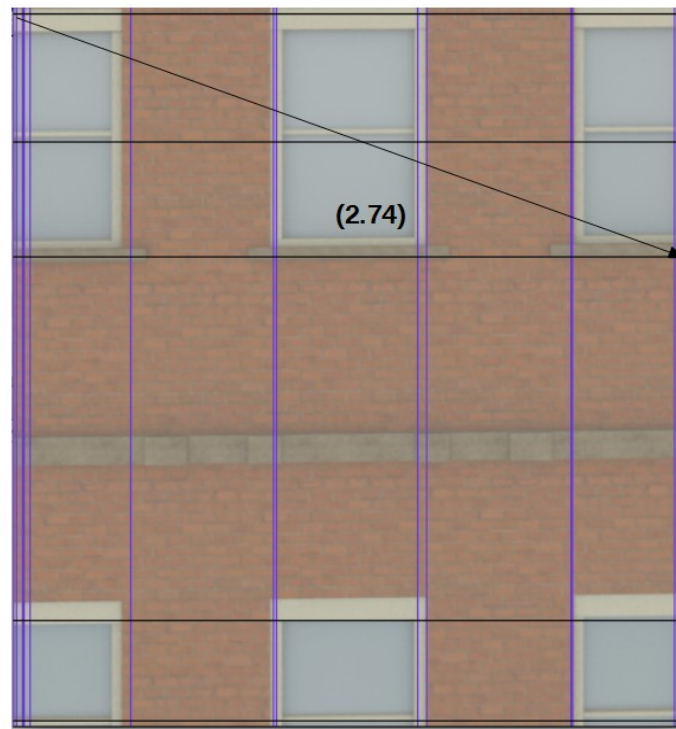


Figure 31: Grid segmented baseline image with width measurements

Model Test: Planar Rectification - Rotation

Test Conditions

This is the rotated case. The quadrilateral retrieved by the previous step (which can be seen in “Test: Vanishing Point – Rotation”) is not fronto-parallel.

Test Overview

Rectification of the image is required. Given the measurement information for the target object and the vanishing point, the line segments detected in the scene must be rectified so that they are vertical (toward y-infinity in the image) and horizontal (x-infinity in the image) respectively. The quadrilateral returned by the previous step fully contains both sets of windows with width covering regions (A – H) and height spanning (2 – 5.5).

Results

While the rectification of the pixel region visually appears to be accurate, the rectification of the vertical (in blue) and horizontal (in black) lines shown in Figure 32b was less successful. A template, Figure 32a, has been included alongside Figure 32b to highlight differences between the grids. For this test, the rectification algorithm fails completely. The rectification algorithm is not reliable. If the quadrilateral found does not require rectification (1.2.A), the resulting grid generated from the planar region is accurate to the source planar region. However, when rectification is required, the resultant grid is not representative of the source planar region. As such, the current rectification algorithm is in need of replacement. Until this has been done, the algorithm cannot be considered to be in a functional state.

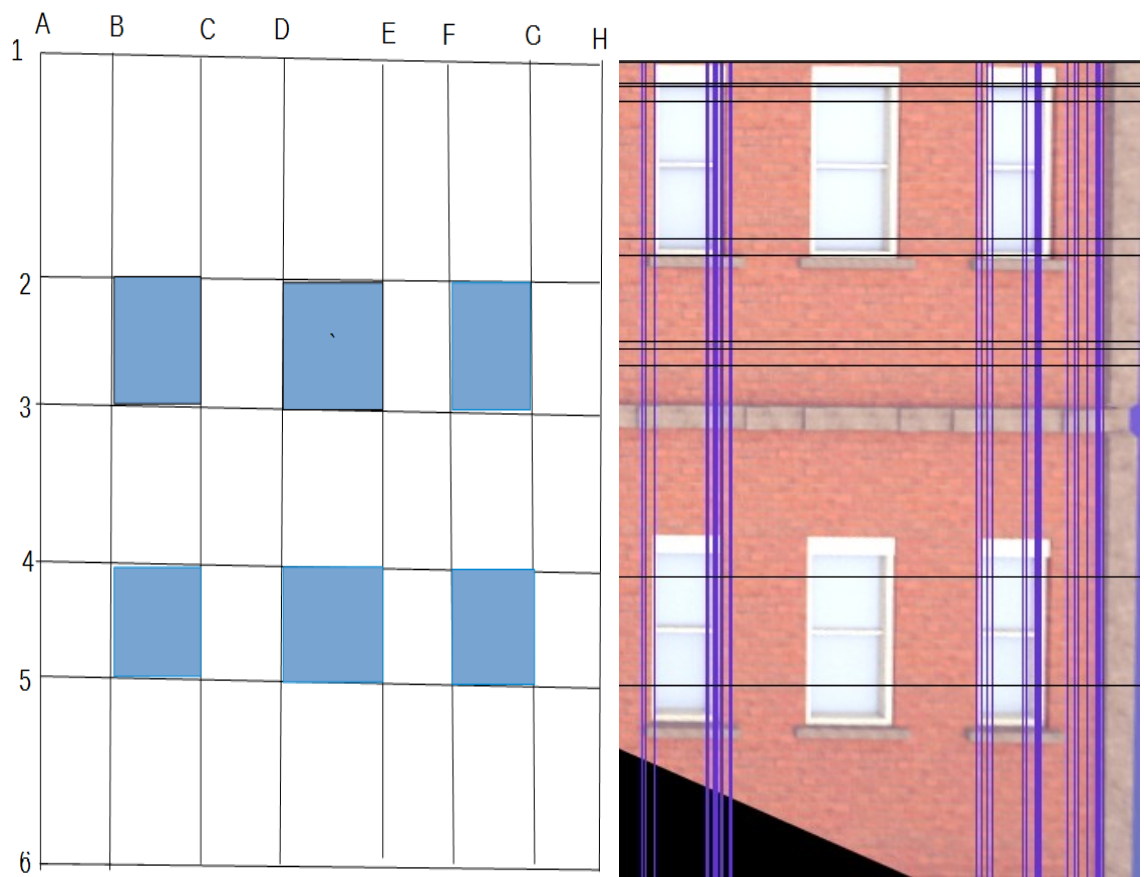


Figure 32 Left: Labeled template; Right: Grid segmented rotated image

Planar Matching

The planar matching compares a potential plane found in the scene to the planes found in the model by their hue histograms. The highest likelihood position of the plane found in the scene when compared to the plane being compared from the model is calculated based on their corner ratios. Based on the pose fitting, a histogram for the potential plane found in the scene is calculated that attempts to identify and avoid the regions of the grid corresponding to windows based on information input in the template saved in the model. This histogram calculated from the scene is compared to a sample histogram saved in the model, via correlation. The tests below demonstrate the histogram comparison on the best case, on the worst case, and the impact of removing or including the windowed regions on the comparison score.

Model Test: Planar Matching - Baseline

Test Conditions

This is the base case. The target object has the same lighting conditions under which the sample texture for the stored model was taken. The target object in this example matches the stored template in terms of the sizing and location of windows.

Test Overview

The sample histogram provided is an exact match to the non-windowed regions. If the windowed regions are correctly ignored, then the theoretical match percentage is 100%. The further from a 100% match the algorithm is, the higher the percentage of windowed regions considered to be non-windowed in the final comparison.

Results

The resultant histogram comparison returned a score of .94, where 1.0 is a perfect match. As expected, when the presented case is easily parsed with identical conditions, then a high match score (> 0.8) is achieved. In Figure 33 below, the hue corresponding matching is shown.

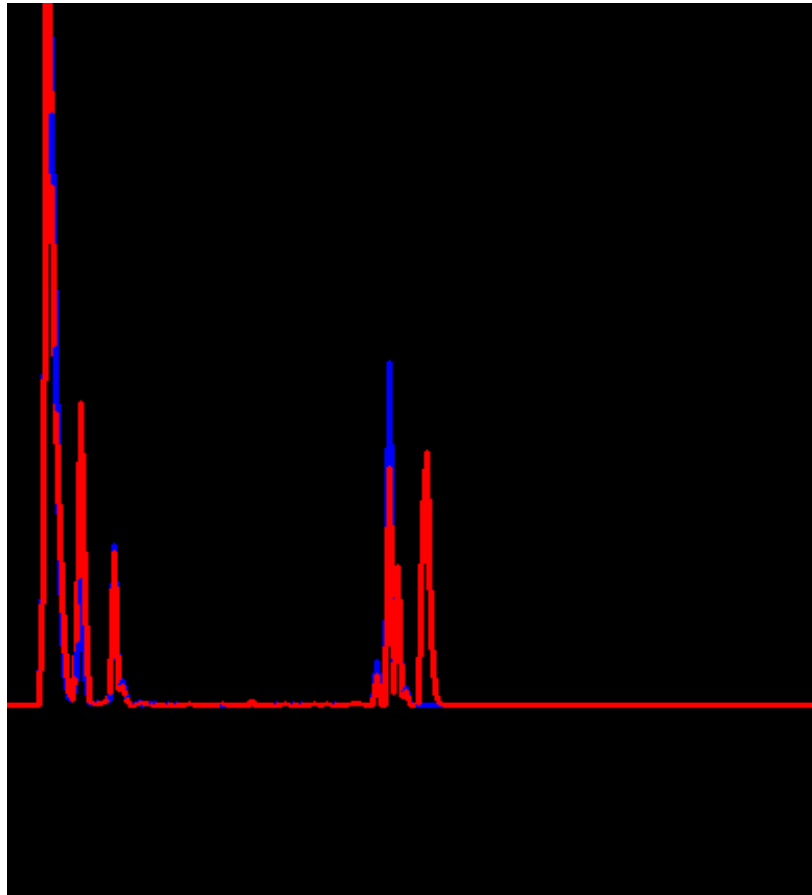


Figure 33: Histogram comparison between the scene (in blue) and the model (in red)

Model Test: Planar Matching – Yellow Lighting Shift

Test Conditions

This is the lighting level color shift test. The target object in the scene was subject to higher levels of yellow light than the lighting conditions of the sample texture from the stored model. The target object in this example matches the stored template in terms of the sizing and location of windows.

Test Overview

This a test for invariance to changing lighting conditions. When performing outdoor recognition accounting for changes in shifting lighting conditions is in important consideration. Unfortunately, as the planar matching currently relies entirely on the hue histogram for the match, the presented algorithm is currently expected to exhibit poor performance on the color light shift test case.

Results

The resultant histogram comparison returned a 0.6 match, a complete mismatch. An interesting result can be seen in the below histogram comparison diagram (Figure 34). The population density for the two histograms are very similar, apart from a shift in the x-axis. As such it seems worthwhile to invest time into testing alternate histogram comparison algorithms that allow for a constant shift.

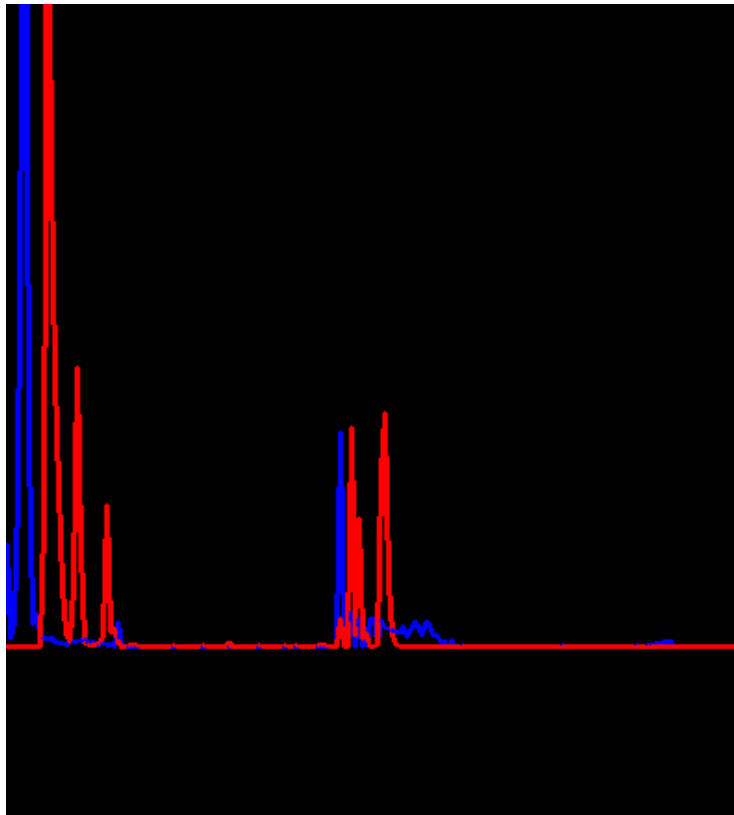


Figure 34: Histogram comparison between the scene (in blue) and the model (in red)

Model Test: Planar Matching – Histogram Degradation

Test Conditions

This test was performed on the base case image. The target object has the same lighting conditions under which the sample texture for the stored model was taken. The target object in this example matches the stored template in terms of the sizing and location of windows.

Test Overview

This is a degradation test. Windowed regions in the image were increasingly marked as non-windowed regions so as to show the degradation in performance when windowed regions are not properly identified. The example used has six windowed regions that account for a total of 65% of the pixel space. The results of the test are displayed below in Table 1.

Measured Values

Windows Removed (Out of Six)	Correlation Score
Six	0.94
Five	0.94
Four	0.94
Three	0.94
Two	0.94
One	0.94
None	0.94

Table 1: Degradation due to window inclusion

Results

No matter the amount of windowed regions removed from the template, the correlation score stays at a constant 0.94. This suggests that the current implementation of the algorithm is unable to remove the windowed regions and the correlation score of 0.94 is the score of the detected plane with its windows included. Properly fitting the grid found in the scene to the template is the result of the vanishing point detection and planar detection components. Until both the vanishing point detection and planar detection components work at the level required of the presented algorithm, the planar matching component will be unable to accurately place the pose of the plane found in the scene to the template in the model for window detection.

DISCUSSION

In constructing the prototype, the goal was to create a system with minimal input to perform object recognition. In line with that goal, a block model was chosen for the world space due to the assumptions that can be made about the scene [12]. Particularly that, given a block world assumption, all objects in the world are cuboid or formed entirely from basic geometric constructs [3]. This can reduce scope of the problem to segmenting the scene into a set of blocks and finding the block in the world.

At the time it seemed that the easiest way to do this would be to locate the planes found in the scene and connect them together to form the blocks. Detecting planarity in an outdoor environment was more difficult than initially expected. Many approaches used indoors are not applicable, specifically those that rely on using the structure of a room itself as a reference point. Structure from motion techniques were then considered but rejected due to the difficulty of implementation and start up time. I wanted the system to be used for recognition, as such I wanted to choose an approach that would decide on the scene geometry first and foremost. At which point this work from Trinh and Jo [44] was found and provided the chosen method for plane discovery.

The goal of this thesis was to demonstrate an approach towards Outdoor Object Recognition that required minimal configuration and input. I cannot reliably

say that this goal has been met. So far the prototype has only been tested on simulated data with mixed results at best. Even with mixed results the assumptions made by the prototype allow the system to be run in most urban environments and the required input information to the system can be gathered with a single image. While the prototype algorithm was constructed, it is not in a state where it can be reliably tested. It is a solid foundation however, and I believe it would be beneficial to continue work on the system.

CHAPTER 5

FUTURE WORK

The future work on this project is split into four independent categories:

- Optimization
- Mobile Port
- Localization and Depth
- Tracking

The presented implementation is too slow for usage in a real time system. Many of the components can be further optimized using more specialized approaches. For this initial prototype, it was not necessary to optimize the individual components, but this must be done before the approach can be considered commercially viable. The feature point extraction and look up would benefit from using dynamic programming to reduce the search space as well as taking advantage of any inherent symmetry prevalent in the scene for faster pose estimation. The system currently has problems with fault tolerance that can be corrected by increasing the maximum number of iterations during the initial line segment extension phase if a minimal number of corner points have yet to be located. Once localization has been performed then it should be used to reduce the search space. During the plane rectification the lines of a plane are rotated geometrically while the pixels are moved via a nearest neighbor

algorithm. This causes matching problems dependent on the degree of rotation. These aspects must all be optimized and fine-tuned before further work can be done.

The system prototype is meant to be used as a framework for mobile OAR applications. As such the system will need to be ported to a mobile device. Due to the amount of freedom concerning interactions with the equipped hardware, Android has been selected as the mobile OS of choice. Android is Java based while the proposed prototype has been written in C++. Aspects of the prototype's software architecture that make liberal use of pointers or the C++ standard library will take time to port to Java.

The prototype is currently hard coded to work from a single location in space in relationship to the target object. The focus of this work has been to present an approach to handling recognition, localization is not in the scope of the current work. This is not unusual for the first version of such systems, as seen in Reitmayr and Drummond's 2006 work [33], which had localization added into the second version of the application the following year. For commercial viability however, localization is a necessity. The most interesting part concerning the localization requirements of the proposed approach is that either a previously developed localization algorithm or a range finder will suffice.

Proper localization will allow for estimation of depth information based on location. One of the biggest problems with the proposed approach is that depth

information is required to accurately rectify the planes from image space into the world space. Given a user's location in space as well as the location to the target object, the respective angle of rotation can be computed and the plane properly rectified. The problem with this approach is that the localization algorithms mentioned in the previous works section all requires a large amount of preliminary data, which the proposed approach wished to minimize.

If a rangefinder is used as opposed to Vision-based localization preliminary data will not be required. Using the depth information a plane can properly be rectified, reducing the potential for mismatches during the feature lookup. The user's location in space can then be calculated based on their possible relation to the object in question. The biggest drawback to this approach, is that mobile phones do not come equipped with rangefinders, requiring an additional sensor. This is contrary to the goal of the work that the presented approach be widely available. If both approaches prove unsatisfactory, alternate algorithms will be investigated.

A third option is available for rectifying the plane that does not rely on either additional hardware to gather depth information or costly localization algorithms. The requirement for this option is that the angle between edges in the world space and the size ratio for line segments in the world space that are not parallel in the image space are known. If these requirements are true, the plane can be rectified with only a loss of scaling information [20]. As this work makes the assumption that planar regions of interest contain grids, the previous conditions are met. All angles between edges in

the grid are 90 degrees in the world space. Additionally, as the grid is constructed from the vanishing lines of two vanishing points, many of the size ratios are known. The drawback to this approach is the loss of scaling information. However, scale invariant ratios are used as the identifying features of the proposed approach, as such this should not be a detriment to the system.

The most important component left to be done is the Tracking Phase of the prototype. This phase will consist of new work with the goal of object tracking based on tracking of the object's vanishing points. The algorithm will track the expected location of the vanishing point in 3-dimensional world space based on its location and frame to frame movement in the image space. One of the decisions to be made is whether the sensors inherent in a modern mobile phone are accurate enough to assist in tracking from 2-dimensional space or if a depth information is needed to track the vanishing point in 3-dimensional space. The other large decision will be on the algorithm to use to locate the vanishing point in the image space. Most vanishing point detection algorithms are specialized for work with roadways. Research will be done on whether there currently exists an approach for precise vanishing point detection that will suit the needs of this algorithm, or whether one will have to be developed. When completed, the tracking phase will perform the frame to frame tracking that will serve as the backbone of the prototype.

BIBLIOGRAPHY

- [1] Angus, J., “What is Yelp Monocle,” May 2013. Retrieved March 17th, 2014 from <http://wearehmc.com/blog/what-is-yelp-monocle/>
- [2] Arth, C. and Schmalstieg, D., “Challenges of large-scale augmented reality on smartphones,” in *ISMAR 2011 Workshop on Enabling Large-Scale Outdoor Mixed Reality and Augmented Reality*, Basel, Switzerland, October 2011.
- [3] Arth, C., Klopschitz, M., and Reitmayr, G., “Real-time self-localization from panoramic images on mobile devices,” in *Proceedings of the 10th International Symposium on Mixed and Augmented Reality (ISMAR)*, Basel, Switzerland, October 2011, pp. 37–46.
- [4] Behringer, R., “Registration for outdoor augmented reality applications using computer vision techniques and hybrid sensors,” in *Proceedings of the 1999 Conference on Virtual Reality*, Houston, Texas, March 1999, pp. 244–251.
- [5] Berg, A. C., Grabler, F., & Malik, J., “Parsing Images of Architectural Scenes,” in *Proceedings of 11th International Conference on Computer Vision (ICCV)*, Rio de Janeiro, Brazil, October 2007, pp. 1–8.

- [6] Canny, J., “A computational approach to edge detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **8**(6):679–698, 1986.
- [7] Ceylan, D., Mitra, N. J., and Zheng, Y., “Coupled structure-from-motion and 3D symmetry detection for urban facades,” *ACM Transactions on Graphics*, **33**(1), 2, 2014.
- [8] Chen, D. M., Baatz, G., and Koser, K., “City-scale landmark identification on mobile devices,” in *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Colorado Springs, United States, June 2011, pp. 737–744.
- [9] Fiala, M., “ARTag, a fiducial marker system using digital techniques,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, San Diego, California*, pp. II:590–596.
- [10] Fritz, G., Seifert, C., and Paletta, L., “A mobile vision system for urban detection with informative local descriptors,” in *Proceedings of the International Conference on Computer Vision Systems (ICVS)*, New York, New York, January 2006, pp. 30–30.
- [11] Gallup, D., Frahm, J. M., & Pollefeys, M., “Piecewise planar and non-planar Stereo for Urban Scene Reconstruction,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, California, June 2010, pp. 1418-1425

- [12] Gupta, A., Efros, A. A., and Hebert, M., “Blocks world revisited: Image understanding using qualitative geometry and mechanics,” in *Proceedings of the 11th European Conference on Computer Vision (ECCV)*, Crete, Greece, September 2010, pp. 482–496.
- [13] Hollister, S., “PlayStation Vita plays augmented reality soccer (hands-on video)”, *The Verge*, January 2012, Retrieved on March 17th, 2014, from:
<http://www.theverge.com/2012/1/12/2701266/playstation-vita-plays-augmented-reality-soccer-hands-on-video>
- [14] Johansson, B., and Cipolla, R., “A system for automatic pose-estimation from a single image in a city scene”. Proceedings of *International Conference on Signal Processing, Pattern Recognition and Applications*, Crete, Greece, June 2002, pp. 68–73.
- [15] Karlekar, J., Zhou, S., and Lu, W., “Positioning, tracking and mapping for outdoor augmentation,” in *Proceedings of the 9th International Symposium on Mixed and Augmented Reality (ISMAR)*, Seoul, South Korea, October 2010, pp. 175–184.
- [16] Kato, H., and Billinghurst, M., Marker tracking and hmd calibration for a video-

based augmented reality conferencing system. Proceedings of the 2nd *International Workshop on Augmented Reality (IWAR)*. San Francisco, California, pp. 85-94.

[17] Klein, G., & Murray, D, "Parallel tracking and mapping on a camera phone," in *Proceedings of the International Symposium on Mixed and Augmented Reality*, Orlando Florida, October 2009, pp. 83-86.

[18] Koutsourakis, P., Simon, L., Teboul, "Single view reconstruction using shape grammars for urban environments," in *Proceedings of the 12th International Conference on Computer Vision (ICCV)*, Kyoto Japan, September 2009, pp. 1795-1802.

[19] Lee, D. C., Hebert, M., and Kanade, T., "Geometric reasoning for single image structure recovery," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Miami, Florida. June 2009, pp. 2136-2143.

[20] Liebowitz ,D., and Zisserman, A., "Metric Rectification for Perspective Images of Planes," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA, June 1998, pp. 482-488.

[21] Maini, R., and Aggarwal, H., "Study and comparison of various image edge detection techniques," *International Journal of Image Processing (IJIP)*, **3**(1), 1–11, 2009.

- [27] Matas, J., Galambos, C., and Kittler, J., “Robust detection of lines using the progressive probabilistic Hough transform,” *Computer Vision and Image Understanding*, **78**(1):119-137, 2000.
- [28] Metrowebukmetro, “Nintendo 3DS hands-on – AR Games revealed and more,” March 2011. Retrieved March 17th, 2014, from: <http://metro.co.uk/2011/03/02/nintendo-3ds-hands-on-ar-games-revealed-and-more-642327/>
- [29] Mörwald, T., Prankl, J., Richtsfeld, A., and Zillich, M., “BLORT – The Blocks World Robotic Vision Toolbox,” in *Proceedings of the International Conference on Robotics and Automaton (ICRA) Workshop Best Practice in 3D Perception and Modeling for Mobile Manipulation*, Anchorage, Alaska, May 2010.
- [30] Nieto, M., and Salgado, L., “Real-time robust estimation of vanishing points through nonlinear optimization,” in *Proceedings SPIE 7724, Real-Time Image and Video Processing 2010*, Brussels, Belgium, April 2010.
- [31] O'Hara, K., “Understanding geocaching practices and motivations,” in *Proceedings of SIGCHI Conference on Human Factors in Computing Systems*, Florence, Italy, April 2008, pp. 1177–1186.

[32] RedCoreStudios, “Apartment Building,” June 27th 2011, Retrieved May 6th 2014, from: <http://www.blendswap.com/blends/view/14155>

[33] Reitmayr, G., and Drummond, T. W., “Going out: robust model-based tracking for outdoor augmented reality,” in *Proceedings of the 5th International Symposium on Mixed and Augmented Reality*, Santa Barbara, California, October 2006, pp. 109–118.

[34] Riemenschneider, H., Krispel, U., and Thaller, W., “Irregular lattices for complex shape grammar facade parsing,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Providence, Rhode Island, June 2012, pp. 1640–1647.

[35] Reitmayr, G., and Drummond, T. W., “Initialization for visual tracking in urban environments,” in *Proceedings of the 6th International Symposium on Mixed and Augmented Reality*, Nara, Japan, November 2007, pp. 161-172.

[36] Robertson, D. P. and Cipolla, R., “An Image-based system for urban navigation,” in *Proceedings of British Machine Vision Conference*, London, United Kingdom, September 2004, pp. 1–10.

- [37] Rother, C., “A new approach to vanishing point detection in architectural environments,” *Image and Vision Computing*, **20**(9):647–655, 2002
- [38] Satoh, K., Anabuki, M., Yamamoto, H., and Tamura, H., “A hybrid registration method for outdoor augmented reality,” in *Proceedings of the 2nd International Symposium on Augmented Reality*, New York, New York, October 2001, pp. 67-76.
- [39] Schaffalitzky, F. and Zisserman, A., “Planar grouping for automatic detection of vanishing lines and points,” *Image and Vision Computing*, **18**(9):647–658, 2000.
- [40] Schall, G., Wagner, D., and Reitmayr, G., “Global pose estimation using multi-sensor fusion for outdoor augmented reality, in *Proceedings of the 8th International Symposium on Mixed and Augmented Reality*, Orlando, Florida, October 2009, pp. 153–162.
- [41] Stamos, I., and Allen, P. E., “3-D model construction using range and image data,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head, South Carolina, June 2000, pp. 531–536.
- [42] Takacs, G., Chandrasekhar, V., and Gelfand, N., “Outdoors augmented reality on mobile phone using loxel-based visual feature organization,” in *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*, Vancouver, Canada, October 2008, pp. 427–434.

- [43] Takeuchi, Y., and Perlin, K., “ClayVision: the (elastic) image of the city,” in *Proceedings of CHI'12: Extended Abstracts on Human Factors in Computing Systems*, Austin, Texas, May 2012, pp. 2411-2420.
- [44] Trinh, H. H., & Jo, K. H. , “Image-based structural analysis of building using line segments and their geometrical vanishing points,” in *Proceedings of SICE-ICASE International Joint Conference*, Busan, South Korea, October 2006, pp. 566–571.
- [45] Ventura, J., and Höllerer, T., “Real-time planar world modeling for augmented reality”, Technical University Graz, Graz, Austria, 2010.
- [46] Widder, B., “Best augmented reality apps,” *Digital Trends*, March 14th 2014, Retrieved Mart 16th, 2014, from <http://www.digitaltrends.com/mobile/best-augmented-reality-apps/>
- [47] Xu, Y., Mendenhall, S., and Ha, V., “Herding nerds on your table: NerdHerder, a mobile augmented reality game,” in *Proceedings of CHI'12: Extended Abstracts on Human Factors in Computing Systems*, Austin, Texas, May 2012, pp. 1351–1356.
- [48] You, S., and Neumann, U., “Fusion of vision and gyro tracking for robust augmented reality registration,” In *Proceedings of the 2001 Conference on Virtual Reality*, Yokohama, Japan, March 2001, pp. 71–78.

[49] Zhao, P., Fang, T., and Xiao, J., “Rectilinear Parsing of Architecture in Urban Environments,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, California, June 2010, pp. 342–349.

[50] Zhou, F., Duh, H. B. L., & Billinghurst, M. (2008, September). Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR. *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality* (pp. 193-202)..

[51] Zillich, M., and Vincze, M., “Anytimeness avoids parameters in detecting closed convex polygons,” in *Proceedings of Sixth IEEE Computer Society Workshop on Perceptual Organization in Computer Vision*, Anchorage, Alaska, June 2008, pp. 1-8.